University of Canterbury

M.Sc Thesis

Cascade Reconstruction Analysis with the IceCube Neutrino Detector

Author:
Joseph McCartin

Supervisor:
Dr. Jenni Adams

Submitted in Partial Fulfilment of the Requirements for the Degree of Master of Science in Physics

July 15, 2009

University of Canterbury, Private Bag, 4800, Christchurch 8140, New Zealand

Contents

List of Figures

Abstract

A study into the reconstruction of cascade like events in the IceCube neutrino detector was performed by utilizing *in-situ* flasher devices. Reconstruction analysis was done with two different flasher settings at each optical module on string 63 at varying depths in the ice. Three different reconstruction algorithms were used to estimate the characteristics of these cascade type events. The characteristics included the estimated vertex position, and the number of photons produced by each flasher. The number of photons produced can be related to the energy for the cascade event via the detailed knowledge of the cascade physics. Results from the analysis show the strengths of using the center of gravity type approaches to estimating the vertex positions of very bright events, and the inability to reconstruct to an accurate position in z which can occur when using more complicated vertex algorithms on the same events. Analysis using the energy reconstruction modules demonstrated the strengths of using methods that take into account the ice properties inside the detector.

1 Introduction

Neutrino telescopes are a part of a rapidly growing research effort into the detection of cosmic rays. These cosmic messengers play a significant role in the understanding of the physics of astronomical sources in our universe. The IceCube Neutrino Detector currently being built in the South Pole is an evolution of the technology developed in neutrino detection over the past few decades. The array of optical sensitive modules deployed in the ice is the largest of its kind, and will provide exceptional insight into neutrino astronomy and the mechanics of distant objects in our galaxy. The detector is designed to be able to detect all three different flavours of neutrinos, as well as many other exotic particles.

Neutrinos play an important part in the Standard Model of particle physics, and make up half of all known leptons, a group of fundamental particles which do not interact through the strong nuclear or electromagnetic forces. Neutrinos were first detected by Clyde Cowan and Frederick Reines in 1956. Their existence had been postulated 26 years earlier by the physicist Wolfgang Pauli to explain conservation of momentum in particle decay. Pauli noticed that the emitted electrons after undergoing beta decay had insufficient kinetic energy and momentum to conserve the balance between the initial and final products. Since conservation of energy and momentum are two of the keystone ideas of physics, he proposed that a very weakly interacting particle (as it could not be detected by his apparatus) was responsible for the discrepancy. The particle was named the neutrino by Enrico Fermi which reflects its neutral charge and small cross section in Fermi's native language, Italian. Initial attempts to measure its mass and momentum were unsuccessful, due to the sensitive equipment needed, and the extremely low probability of interaction. In order to obtain the physical properties of the neutrino, it was initially proposed to observe the recoil energy and momentum of an atom undergoing inverse beta decay, as the values would solely depend on the emitted neutrino [1]. Such experiments were not directly detecting the neutrino itself, and were only performed to put rough limits on the neutrinos properties.

In the 1950s, the two physicists Clyde Cowan and Frederick Reines used a nuclear reactor to bombard protons with electron neutrinos. In the reactor, neutrinos were created from the beta decay of heavy nuclei, and then interacted in the nearby proton source via inverse beta decay. This produced a detectable positron and neutron, which confirmed the existence of the neutrino as a part of the overall reaction (due to

conservation of lepton flavour). The results were first published in the paper *Detection of the Free Neutrino: a Confirmation* [2], which later won Frederick Reines the 1995 Nobel prize for physics [3]. Shortly after Reines' discovery, in 1962 the muon neutrino was discovered by Leon Lederman, Melvin Schwartz and Jack Steinberger [4] who were observing pion decay in a synchrotron accelerator. With the knowledge of the electron and muon neutrinos, and the discovery of a third lepton in 1975 *tau* (τ) [5], particle physicists speculated the existence of a third neutrino, to pair with the recently discovered charged particle. The final neutrino was discovered as part of the DONUT collaboration at Fermilab in the summer of 2000 [6].

Due to the large number of neutrinos produced in the cosmos, and to their ability to travel very long distances without interaction, neutrinos can offer us unparalleled insight into the workings of processes such as supernovae and various accelerating regions throughout the universe. One of the first astrophysical uses of neutrinos after their detection, was as a final confirmation of the nuclear fusion process in our own sun. Electron neutrinos are a by-product of helium formation from the fusion of hydrogen nuclei, and are created in vast quantities in the Sun's core.

In the late 1960s, it was discovered by the astrophysicist Raymond Davis, Jr. that the observed number of electron neutrinos was one third of that predicted (by the physicist John N. Bahcall as a part of his theories on stellar nuclear reactions). This deficit in the expected number of electron neutrinos became known as the Solar Neutrino Problem. After a change in the expected ratio of muons to electrons was observed during studies of atmospheric muon neutrinos by the Super-Kamiokande collaboration [7], a possible solution to the Solar Neutrino Problem was suggested via the the incorporation of neutrino flavour oscillations into the solar neutrino flux models. As a consequence of what is known as the Mikheyev-Smirnov-Wolfenstein *(MSW) effect*, electron neutrinos created in the sun oscillate into the two other flavours of neutrinos due to the interaction between the neutrino and the matter it propagates through [8]. For the neutrino energies produced by the Sun, the distance to earth is long enough for the oscillation effect to equally reduce the number of electron neutrinos by two-thirds. When detection of all three neutrino flavours was possible, the sum of the observed flavours corresponded to the original estimate of the number of electron neutrinos produced in the core of the Sun. By 2001, the SNOW neutrino detector in Canada had observed a number of neutrino events confirming predictions made by the original solar flux models, by being able to detect the sum of all three neutrino flavours [9]. The success of the predictions of the solar neutrino flux, and the solutions to the Solar Neutrino Problem were recognised by the Nobel committee, and in 2002, Raymond Davis and the physicist Masatoshi Koshiba were each awarded the Nobel Prize for their contributing efforts [10].

This thesis looks at the methods used in analysing neutrino induced cascades, which are an important part of the large amount of data generated by IceCube each year.

By studying the performance of these methods on controllable input data; such as the flashing of LEDs on each optical module, the strengths and weaknesses of each method and associated algorithms can be assessed.

The first chapter in this thesis introduces the IceCube detector, and describes the properties of the ice that surrounds the optical modules. Lastly, the chapter describes the systems used in data collection and processing. Chapter 3 details the physics of neutrino interactions, and the two types of cascades that occur in IceCube. The production of Cereknov radiation from charged leptons is also mentioned, and how the scattering and absorbing elements in the ice distort the light distributions in each observed event. The software used in IceCube is described in Chapter 4, specifically; the tools used for simulating neutrino and flasher data, and the methods used in reconstructing the data. The methods that were used are described mathematically, as well as their position in a typical reconstruction chain. Finally, Chapter 5 covers the reconstructions performed on data generated by the flasher LEDs that are embedded on each optical module in the ice. The chapter covers the analysis on two series of data runs that were each generated with different flasher brightnesses.

2 IceCube Detector

The IceCube project is an ice-based neutrino detector residing at the geographic south pole in Antarctica. It is the continuation of an earlier project called *AMANDA*, which was built to test the feasibility of using ice-based optical modules for neutrino detection. The success of *AMANDA* over its period of operation, allowed it to be able to constrain many models for astrophysical neutrino production. Construction first started on IceCube in 2005 with the lowering of the first of 80 strings into bore holes drilled into the ice by hot water drill. The planned 80 strings will form a hexagonal pattern for a total volume (instrumented) of 1km^3. At the time of writing, the detector had 40 operational strings installed, each with 60 digital optical modules (DOMs) suspended in the nearly optically-transparent ice. By the end of the summer at the South Pole in 2009, the detector will have 16 or 18 more strings installed, for a maximum total of 58. As well as housing a photomultiplier tube (PMT) for the detection of photons, the DOMs contain various electronic apparatus used for calibration tests.

When a charged particle moves through the ice, the particle produces visible light which can be detected by the DOMs. If a PMT detects a photon, the DOM transmits the waveform information of the hit up to the surface station resting on each string, which is then sent to the counting house where data is recorded from all strings. All of the recorded hits in a certain time window are registered as a single event, which is saved if the number of total hits in the detector meets a certain lower limit. An event is a collection of all of these recorded hits in the trigger time window. Events are stored as a map of all hits on all activated strings in three dimensional space. By looking at the recorded hits in each event, it is possible to apply a number of 'reconstruction' routines on the data, in order to identify certain physical parameters of the particle that caused the particular event. The IceCube detector records and stores vast quantities of this information, so it can be later used by the collaboration for analysis. As well as from candidate neutrino sources, data can also be triggered by *in-situ* devices within the ice. For a study in cascade detection, both flashing LEDs installed on each DOM and calibrated laser sources currently deployed on two strings called *Standard Candles*, are available as an effective light source, that creates a distribution similar in shape to cascades.

This study used the 40-string detector array, with the string layout shown in figure 2.1. Currently, in the middle of the 2008/2009 deployment season, the detector has an additional 10 deployed strings, which are inactive until the freezing of the

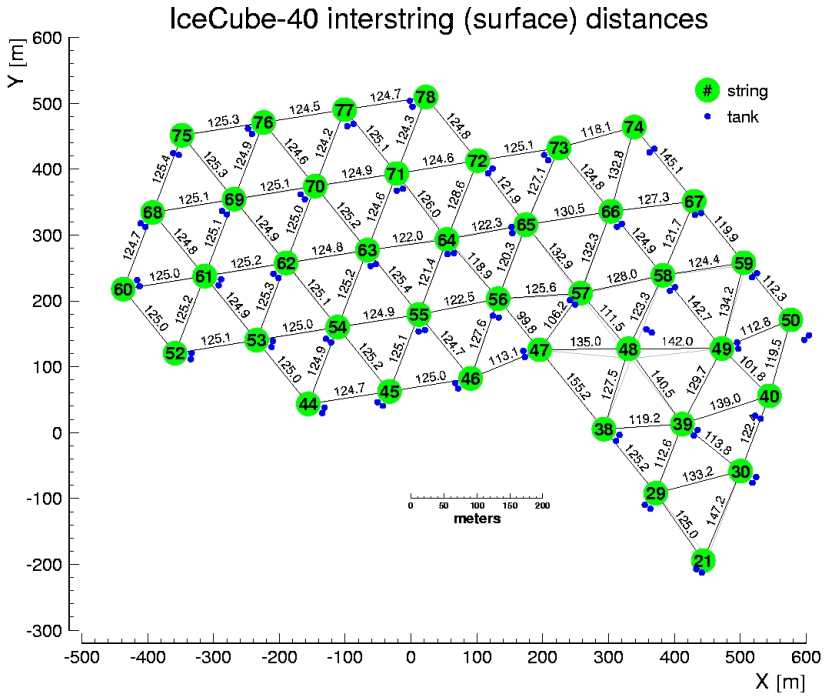

Figure 2.1: IC-40 string deployment in IceCube (2008)

water surrounding them has completed.

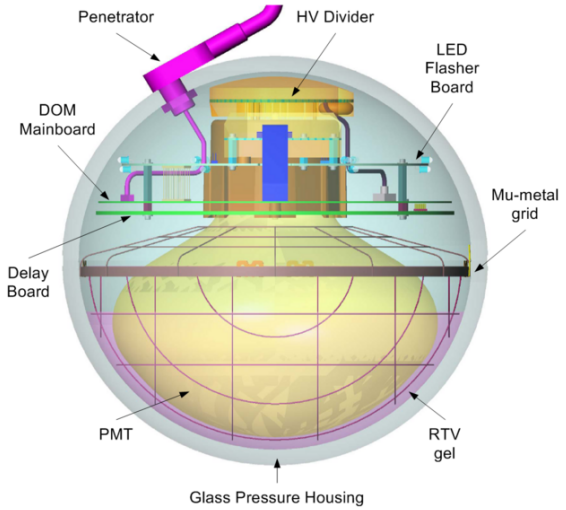

Figure 2.2: Cross section of a DOM deployed in IceCube

2.1 Data Acquisition

The PMT on each DOM has been calibrated in a specific way so that its *Quantum Efficiency*; its ability to detect photons, peaks at the energy of Cerenkov radiation most frequently seen in the detector. This photon energy corresponds to a wavelength of 405nm. When the PMT on a DOM receives optical photons, it records the amount of photoelectrons produced in the PMT over the period of detection. The generated current is passed through a gain to obtain the voltage in the PMT as a function of time. If the wavelength of all of the photons is assumed to be constant at 405nm, the number of photons detected by the PMT can be found via the number of detected photoelectrons. During bright events such as high energy cascades however, many Cerenkov photons can be incident on the DOMs close to the origin of the light, in a small amount of time. This causes saturation of the DOM electronics, which reduces the ability of the DOM to detect how many photons it received over the time interval. This is shown clearly in figure 2.3, where for expected voltages greater than 3500 mV, a smaller measured voltage is recorded. As a consequence, less photoelectrons are recorded by the DOM electronics, and therefore the photon count will be lower than expected for a given event brightness. To counter this saturation problem, an array of different data collection subsystems are built into each DOM.

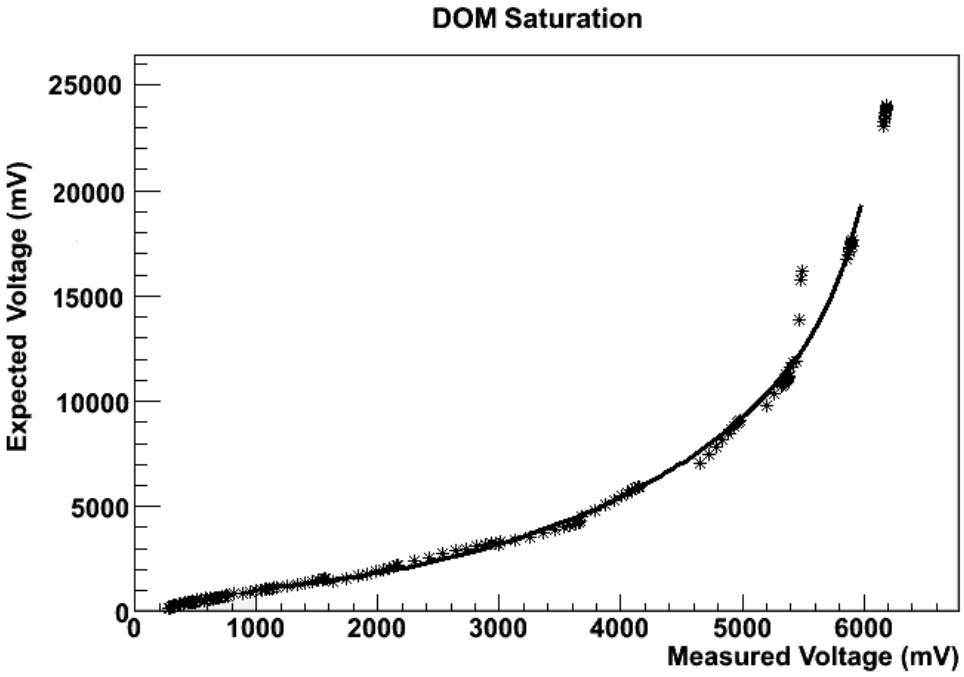

Figure 2.3: Saturation of the IceCube DOM (39,13) from a flasher event centered at DOM (39,15). The relationship between expected voltage and measured voltage is approximately linear up to the saturation limit, at 3500 mV.

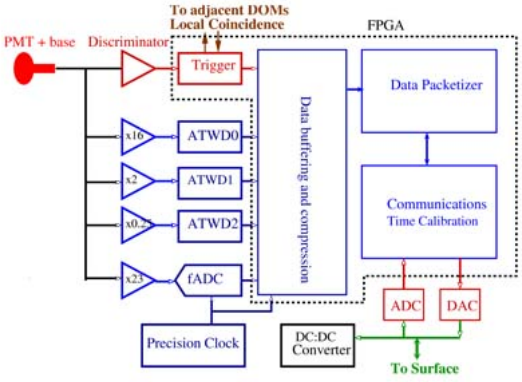

Figure 2.4: Graphical layout of the location of the FADC and ATWD subsystems on the mainboard of each DOM

2.1.1 FADC

The fast Analogue-to-Digital Converter (FADC) is a data-aquisition subsystem on the DOM mainboard which provides accurate information on the arrival times of photons that the PMT receives. The FADC subsystem runs at an effective sampling speed of 40MHz, and records the arrival times of photons over a $6.2\mu s$ period [11]. A long period of aquisition allows for the detection of late photons, which can result from heavily scattered light from candidate events. Because only the arrival times of Cerenkov photons are of importance to the FADC subsystem, saturation of the channel is not taken into consideration.

2.1.2 ATWD

The Advanced Transient Waveform Digitizer (ATWD) subsystem provides information on the charge of the waveform each DOM receives, as group of photons enters the PMT. The ATWD has an effective sampling rate of 300 megasamples per second, which is significantly faster than the FADC subsystem, for the purpose of having increased accuracy during the recording of waveforms from a candidate neutrino event. In order to accurately record the charge information of the incoming signal, three ATWD systems were introduced onto the mainboard on each DOM. ATWD records the waveform over three channels running in parrallel, each operating at different gains ($1/4, 2$ and 16). The ATWD subsystem is capable of detecting waveforms 400ns in length [11]. If a group of photons causes the ATWD0 bin to saturate, the data is recorded from the ATWD1 channel, operating at a higher gain. If this bin also saturates, data is recorded from the final ATWD2 channel. A fourth ATWD also exists, which can record the electronic signal sent to the flasher LEDs on the

flasher mainboard.

2.1.3 Noise Reduction

Because the PMTs on each DOM are very sensitive to hits from single photons in the ice, this sensitivity makes the electronics on the DOM susceptible to random noise hits. In order to reduce the amount of recorded data triggered by noise hits, a number of local coincidence settings are built in to the IceCube DAQ. These coincidence settings are designed to reduce the amount of noise recorded by the detector by introducing a requirement, where an additional hit must be received on two of the neighbouring DOMs within a given time window. This is called the local coincidence (LC) span 2 setting.

2.2 Ice Properties

While the ice surrounding the DOMs is optically transparent, it will scatter the Cerenkov light after a few tens of meters depending on the exact wavelength of the light. The primary reason for scattering is trapped air bubbles and dust particles which get deposited at certain layers in the ice. Research done in 2005 on the optical properties of the glacial ice found at IceCube found that the four most common particles responsible for scattering were trapped sea salt, mineral grains, soot and liquid acid drops possibly formed by sulphates in the atmosphere [12]. Studies of ice samples done with light pulses from a dye laser found that the number of air bubbles trapped in the ice decreased with depth, but at 2.1km a 100m thick band of dust could cause a hindrance on detection [13]. This is shown rather strongly if DOMs near the dust layer produce flasher events, where the produced light distributions are heavily affected by the absorption.

A study done earlier by the AMANDA collaboration [14] found that the absorption length for 515nm visible light in ice at a depth of 1km is 59 ± 3 metres. This length is comparable with pure heavy water used in the Kamiokande detector, and better than for ice formed in a laboratory. At this depth, the absorption length is roughly half of the average inter-string distance, and decreases on average as a function of depth in the detector. Light absorption reduces the ability of the detector to observe events which occur outside of the detector, as all of the light from interactions occurring outside of the range of the DOMs will be absorbed. Scattering is also relevant to this study as it can result in muon tracks appearing more diffuse and cascade-like.

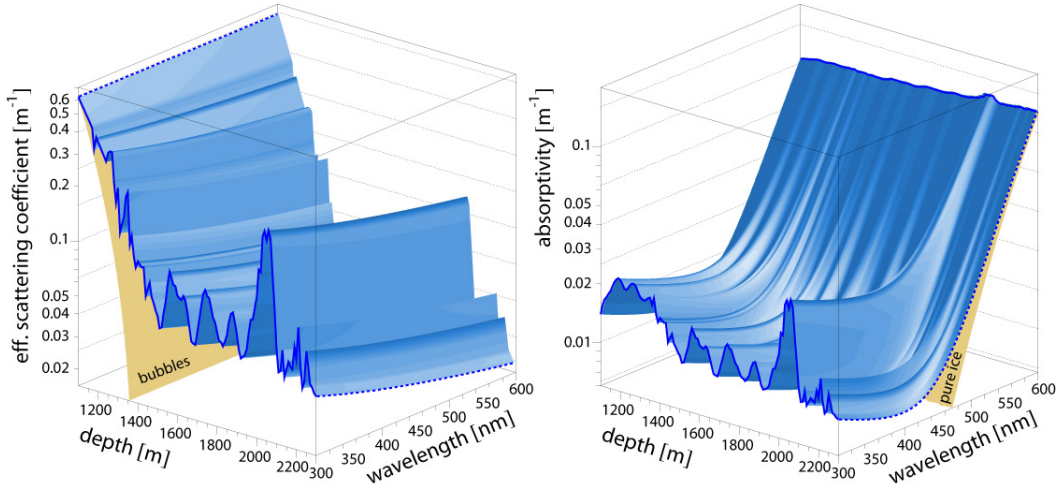

Figure 2.5: Scattering and absorption of light as a function of depth and wavelength over the optical range of the DOMs in IceCube

2.3 Data Processing and Filtering

Large amounts of data processing is essential at low levels, in order to filter out unwanted data, and to isolate candidate signal events. Since the data bandwidth transmitted via satellite is limited, processing is done at the South Pole in a *Counting House*, which handles much of the low level filtering.

Filtering of the recorded data is done at IceCube in order to select events which are useful for different studies of the flux of neutrinos expected at the detector. Filters use a range of IceTray projects in order to tag events according to their estimated type, through various reconstruction methods. This enables different parts of the collaboration to look at certain types of events that may be useful for their study, while remaining a background for others. An example of this is with candidate cascade data taken with the detector. Muons pass through the detector at a rate higher than the expected flux of cascade events, therefore filling any data taken over the running period with muon tracks. Since muons are very common, they act as a large background signal, and so must be purged from the data before the cascade reconstruction algorithms can be applied meaningfully. This is even more important for cascades, as the stochastic nature of Bremmstrahlung emitting muons can produce false positives to the cascade reconstruction routines.

3 Physics of Neutrino Interactions

This chapter covers the theory side of neutrino interactions with matter, and how the characteristics of the events at the vertex make it possible for the optical modules deployed in the ice to detect them. The next section mentions the two different types of electroweak interactions between the neutrinos and matter, and how each neutrino can produce detectable charged leptons, through their production of Cerenkov light. The final section describes the production of Cerenkov light itself, and how the light distributions of each event show characteristics unique to each flavour.

3.1 Vertex Interactions

When neutrinos interact with matter, they do so via the exchange of Z^0 or $W^{\pm}$ bosons, the carriers of the electroweak force. In the IceCube detector, interaction is most likely to occur with the nuclei of either a hydrogen or oxygen atom which make up the majority of the ice that surrounds the detector. Where the interaction involves either a Z^0 or a $W^{\pm}$ boson with the nucleon, the neutrino will undergo a Neutral Current (NC) or Charged Current (CC) interaction respectively.

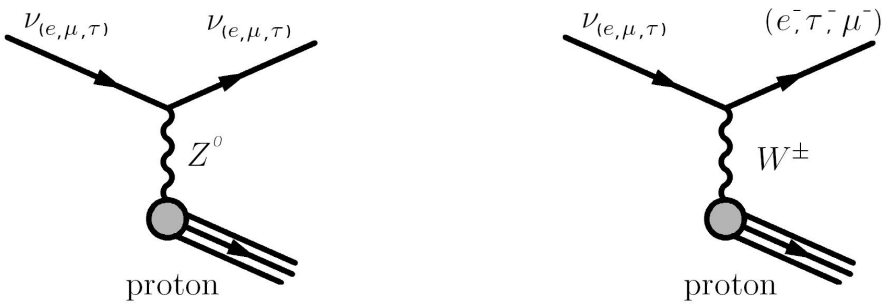

Figure 3.1: The Neutral and Charged Current interactions of a neutrino with a proton, the three arrows emanating from the proton indicate a hadronic cascade.

Figures 4.3.2 and 4.3.2 show the two types of neutrino interactions with matter. In the NC case, the neutrino is 'scattered' by the interaction with the matter in the medium, transferring some of its energy to the nucleon. After the interaction, the

neutrino's trajectory changes very little from its original path [15]. When a neutrino interacts via the CC interaction, a lepton of identical flavour is produced at the vertex.

Since the electroweak force acts over a very small range, neutrinos must be in close proximity to a nucleon in order to interact. Because of this, the cross sections for the NC and CC interactions are very small. For IceCube, the cross sections for the CC interactions are on average three times larger than those for the NC interactions [16]. This means that the probability of a charged lepton being created at the vertex is three times higher than the probability of a neutrino being scattered at the neutrino interaction vertex.

3.1.1 Hadronic Cascades

For the two types of neutrino-matter interactions, the electroweak boson exchange can transfer a large fraction of the neutrino energy to the quark at the vertex. This energy imparted upon the proton gets transferred to the nucleons three quark constituents, which produce a hadronic cascade with the surrounding matter. It is called a hadronic cascade because the initial interaction constituents are strong force interacting hadrons. If a neutral current interaction is the only detectable interaction in the detector, only a hadronic cascade would be observed via the produced Cerenkov light, as the scattered neutrino would be very unlikely to interact again in the observation volume. All three neutrino flavours produce hadronic cascades at the vertex of the NC/CC interaction.

3.2 Creation of Charged Leptons

3.2.1 The Electron

The muon and tau leptons, having larger rest masses than the electron, have a greater ability to propagate large lengths through the ice. An electron will lose a large majority of its energy at a point close to the vertex, and can cause what is known as an electromagnetic cascade, which will be covered in more detail in the next section.

3.2.2 The Muon

A muon will typically travel long distances through the ice, producing Cerenkov radiation as it propagates. A TeV-energy muon will produce a track length of Cerenkov light of roughly 3km in length through the ice [17]. Once the muon passes below the minimum Cerenkov energy threshold, it will no longer be visible by photon production, and will then most commonly transfer the remaining energy into the lepton's

decay by-products. In rare cases, the muon can effectively 'stop' in the ice, being captured by a nearby atom. One other characteristic of the muon which is of importance to a study of cascades is the muon's stochastic nature of energy loss. A muon loses energy along its track randomly throughout the detector, due to multiple scattering off nearby matter, and Bremsstrahlung radiation processes. Due to the large spacing between strings in IceCube, a muon may leave a distinct high energy bremmstrahlung emission that could be mistaken for a cascade signal, if the Cerenkov light from the muon track itself is not detected. Since many muons are produced in the upper atmosphere above the detector, the tracks caused by these leptons represent the largest background to any study in cascade detection with IceCube.

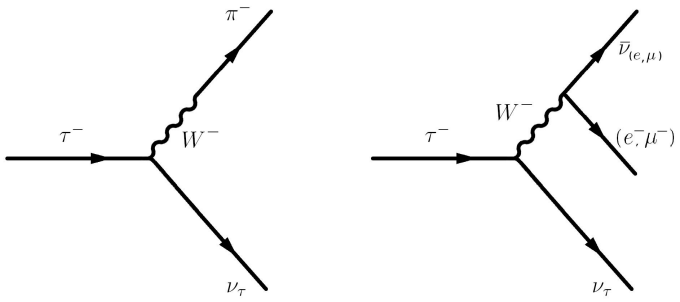

Figure 3.2: Tauon decay into a pion and other leptons through the W^- boson.

3.2.3 The Tau Lepton

The tau lepton, with a much smaller half-life than that of the muon, decays faster, resulting in a significantly shorter track length. The tau decay leads to another hadronic cascade, with a number of different decay products. Two illustrated examples are in shown figure 3.2. Pion creation in the left, or electron or muon creation in the right diagram in figure 3.2 make up the three most common decay modes of the tau lepton. Pion formation accounts for 11% of all tauon decays, while the second most common possibility (antineutrino and lepton formation) account for 20% of all decays each [18]. Because it is very likely to decay shortly after creation, the tau lepton may produce a distinct 'double bang' if the two hadronic cascades are observed in the detector [19]. The track length of the tau lepton is directly proportional to the energy of the incident neutrino. The higher the energy of the tau lepton, the greater its lifetime will be in the detector due to relativistic time dilation. When the energy of the tau lepton is greater than 1PeV, the track may be several tens of metres long [20].

3.2.4 Electromagnetic Cascades

In the case of a charged current interaction, the charged lepton produced at the vertex has a much higher probability to interact inside the detector compared to the neutrino itself. Interaction probability of the charged lepton varies according to the lepton flavour, as each flavour has different rest masses. All three charged leptons will produce Cerenkov radiation along their tracks, which is detectable by the DOMs. The electron has the shortest interaction length of all three flavours. The charged lepton will quickly produce its own electromagnetic cascade from a large number of Bremsstrahlung radiation and pair production events (figure 3.3), this continues until the constituents of the cascade fall below the cut-off level for pair production. Electromagnetic cascades are most common to the electron, due to the electron's small track length, which cause it to transfer a large amount of energy to the surrounding matter in a small area. The electron produces a cascade of repeated pair production and Bremsstrahlung events, which can be described as a cylindrical distribution of forward moving leptons. A muon may also produce a cascade in this manner, if enough energy is lost to the surroundings via a stochastic Bremsstrahlung emission along the muon's track. For an incident neutrino energy of 100TeV the cylinder for an electromagnetic cascade would have dimensions of width measured in tens of centimeters, but a length of about 8.5m [17]. The charged particles in the cascade will each emit Cerenkov radiation, until they fall below the speed of light in the medium. In the non-relativistic limit, the total length of the cascade will be directly proportional to the number of bremmstrahlung/pair production events.

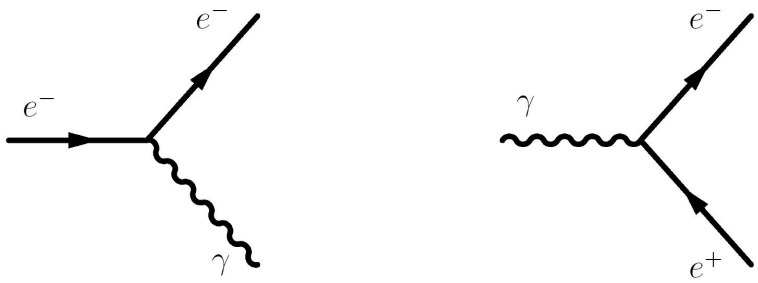

Figure 3.3: Bremsstrahlung and pair production in an electromagnetic cascade.

In the absence of scattering, the Cerenkov radiation produced by the cascade would only be detected in locations forward of the cascade, allowing accurate measurements on the cascade direction, and therefore the zenith angle of the incident neutrino. Scattering and absorbing elements in the medium such as air bubbles and dust would affect the observed light distribution by diffusely scattering the Cerenkov light after only a few metres, therefore making the light distribution look spherical at distances

far enough from the vertex.

If the energy of the incident neutrino is high enough, the relativistic effects can no longer be ignored, as the cascade will be elongated due to the suppression of bremmstrahlung production due to the multiple scattering of the energetic electron. This effect was first noted in 1953 by the physicists Landau, Pomeranchuk and Migdal, and is now known as the LPM effect[21]. These elongated cascades will produce a light distribution throughout the detector noticeably different from that of the lower energy ones.

3.3 Detection of cascades

3.3.1 Cerenkov Radiation

When a charged particle enters a medium where its velocity exceeds that of the reduced speed of light c/n (where n is the refractive index of the medium), it produces photons of electromagnetic radiation, which radiate in the forward direction of travel. This radiation peaks at around 400nm, and increases in wavelength in energy for slower moving particles. This visible radiation can be detected with light sensitive photomultiplier tubes that have been deployed throughout the IceCube detector.

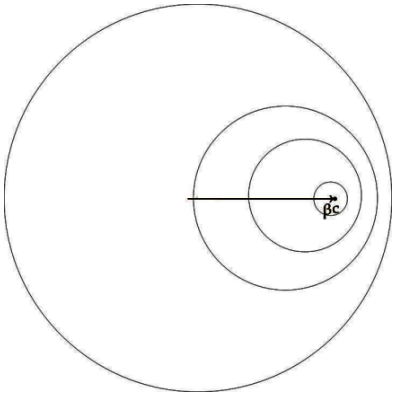

Figure 3.4: Electromagnetic wave pattern from a charged lepton when $v < \frac{c}{n}$

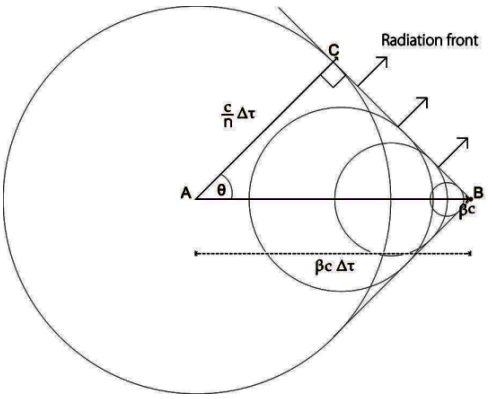

Figure 3.5: Cerenkov formation from a charged lepton when $v > \frac{c}{n}$

Figure 3.4 shows a particle with velocity v entering a medium of refractive index n, traveling from point A to point B. In the case of figure 3.5 the particle has exceeded the speed of light in the medium, and will travel a distance $AB = \beta c \cdot \Delta\tau$. The light from each pulse spreads radially according to Gauss's theorem $\rho = c\nabla \cdot \vec{E}$, but reaches the shock front at a minimal distance $AC = \Delta\tau \cdot (c/n)$. The light forms from the build up of electric fields radiating in all directions at the wave front. Since ACB

forms a right angled triangle at C, the angle of the shock front is

$$\cos\theta = \frac{1}{\beta n} \tag{3.1}$$

The refractive index of light in ice is 1.31. In the case of a highly relativistic muon traveling through the ice at speed $\sim c$, then the Cerenkov light cone would have an opening angle of $\theta \simeq 40.2°$. Since this radiation is emitted in the forward direction only, a moving source of Cerenkov radiation such as a muon, leaves a distinct track which can be detected by an array of photomultiplier tubes.

3.3.2 Flavour Identification

As discussed earlier, only a muon-neutrino will produce a long range lepton which travels a significant distance in the detector. Thus long track events can be associated with muon-neutrinos. On the other hand, cascades can be caused by any neutrino flavour. Due to the fact that all neutrinos can produce cascades from NC and CC interactions, it is practically impossible to make a one to one correspondence between a cascade-like event and the initiating neutrino's flavour. However, a ratio of track and cascade events can be used to investigate the ratio of neutrinos flavours observed in IceCube. This flavour ratio can be further investigated if identification of the distinct 'double bang' tau lepton events is proved possible. Figures 3.6 - 3.8 illustrate the three light distributions caused by events characteristic to each lepton flavour, for increasing photon hit times (red through to blue).

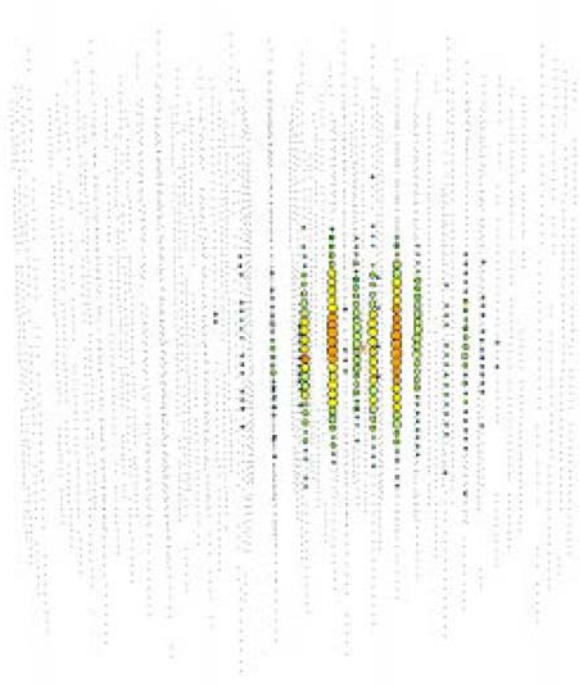

Figure 3.6: The light distribution from a simulated electromagnetic cascade caused by an electron neutrino interacting in the detector.

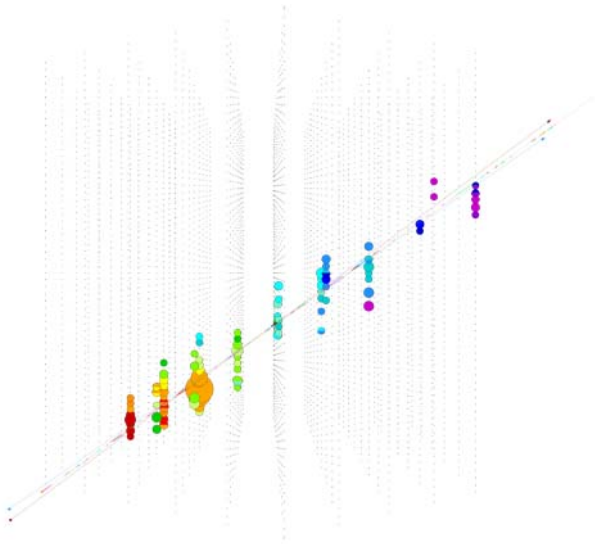

Figure 3.7: An upward going muon causing a visible Cerenkov track detected by the DOMs as the muon propagates along its path.

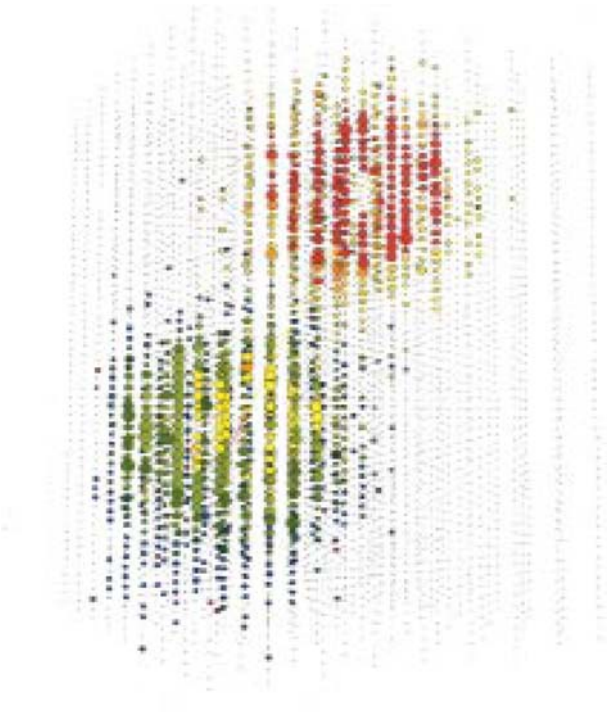

Figure 3.8: Light distributions from two simulated hadronic cascades caused by a 'double bang' tau lepton event.

4 Simulation and Reconstruction Methods

This chapter introduces *Icetray*, the software used for analysis and simulation in the IceCube project. The chapter also describes the theory behind the reconstruction methods used for cascades, and how the techniques may be used to analyse data taken from Monte-Carlo and flasher studies.

4.1 Icetray Overview

Icetray is the software project developed by the IceCube collaboration for both the Monte Carlo generation and analysis of data. Icetray uses an array of modules largely written in C++, although some are written in FORTRAN and Java. *Icetray* is split into different meta-projects, each used for different aspects of the IceCube project. *IceSim*, is used to generate Monte Carlo data simulating neutrino or other physics events which might be seen in the detector, and will be briefly covered in the next section. *IceRec* handles the reconstruction of events observed in the detector. This is used to obtain estimates on parameters such as the zenith angle and energy of events seen by the DOMs. *Icetray* can also be used to study the performance of the detection devices themselves which is important for calibration purposes.

The *IceTray* modules are used in conjunction with a Python steering file. This provides a common interface for the execution of each module on each recorded physics frame. The steering file and general layout is similar for both the simulation and reconstruction sides of the software.

4.2 Simulation Production

The simulation of any physical event is done as a multi-leveled process, with each step calling on different modules from *IceTray*. A standard set of the commonly used modules is updated into a meta-project on a regular basis, and is released under the name of *IceSim*. The multi-leveled structure of the code helps with versatility. A neutrino generator for example can be replaced by a flasher-generator in order to simulate light distributions from the LEDs on each DOM with little modification to

existing code. The Monte Carlo works by creating physics frames within a data file in a similar form to how the data is recorded at the detector at the South Pole.

Simulation and reconstruction output can be viewed by the *glshovel* and *dataio-shovel* programs. These programs provide an interface to view the contents of an *i3* file - the standard container for project files that contain the GCD and physics frames used by the collaboration. The *glshovel* program provides a graphical user interface with a three dimensional image of the detector and the string arrangements. It is possible to select any processed dataclass from each frame, such as the positions of the estimated vertex (in the case of reconstructed data) or the true vertex (for the Monte Carlo data). It is also possible to look at the waveforms received on each DOM for an event. Overall, this can provide a useful way to look at the data, as it displays the data from each event in a three dimensional form, similar to how one would observe them standing at various positions inside the detector. *Dataio-shovel* is an interface which can display the XML raw data of each dataclass in each frame. This is most useful to find information such as the flasher brightness setting on a particular DOM, by looking at the information dataclass created by the ATWD3 bin used by the flasher.

4.3 Reconstruction

When reconstructing each physics frame, steps of increasing complexity are used. This is necessary in order to filter out the large number of background events. Because of this, the modules used at lower levels are fast algorithms, and so can process a large number of events quickly. The first such step in finding an estimate for the event parameters is called the 'first guess', which is then used to seed later, more complicated algorithms also known as Likelihood Maximizers. Many of the reconstruction steps were originally designed for AMANDA and then applied to IceCube. The total reconstruction chain is shown schematically in figure 5.1.

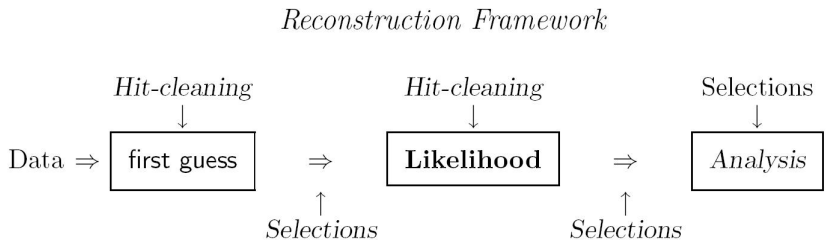

Figure 4.1: Schematic principle of the reconstruction chain [22]. Selections can be made after the first guess and Likelihood sections in order to filter out unwanted data from the analysis.

4.3.1 First Guess

A first guess algorithm gives a quick estimate on some of the vertex parameters in order to provide a seed for the Likelihood Maximizers. The first guess methods are designed to be simple and fast, as many of them are run on data that is used for analysis over the entire IceCube project.

Tensor of inertia

This is a method used by the *phys-services* module that gives each DOM hit a virtual weight, which is then used to work out a center of gravity (COG) [23]. The algorithm weights each DOM with the amplitude of its hits a_i, and then takes the product with the DOMs position r_i for each hit DOM in the event. The origin of the sum of all components is the center of gravity of mass distribution

$$C\vec{O}G = \sum_{i=1}^{N} a_i^{\omega} \vec{r_i}. \tag{4.1}$$

A varying amount of weighting can be given to the amplitude of the hits, using the exponential coefficient ω. This coefficient is known as the *AmpWeightPower*, and is normally set at 0 or 1. The tensor of inertia is found using

$$I^{k,l} = \sum_{i=1}^{N} a_i^{\omega} (\delta^{kl} \vec{r_i}^2 - r_i^k r_i^j). \tag{4.2}$$

The Tensor of Inertia method works well in identifying and reconstructing cascades, as the cascade light distributions are roughly spherical. This method can also be used to reconstruct muon tracks, since the smallest eigenvalues of the tensor will correspond to the longest radial axis. The values for the axes will be roughly similar in all three directions for a spherical cascade, distinguishing it from a track. The Tensor of Inertia method is versatile for a range of different signals and so therefore it is used in the low level IceCube filters to help distinguish candidate cascade events from track type events.

CFirst

CFirst uses a variable-weighting approach to estimate the vertex information of a cascade-like event [24]. Similar to the calculations done in the *phys-services* module, it finds the vertex based on the center of gravity approach to the hit distribution. CFirst also has the ability to set the amount of weighting per DOM as with the Tensor of Inertia routine, however for bright events the weighting makes the algorithm slow, due to the large number of hits each DOM may receive. CFirst makes

use of the time residual parameter t_{res}, defined as the difference between the time of the received hit and the expected time of a hit caused by direct light. The difference in times arises from the effects of scattering of light in the detector medium, and higher values of t_{res} are expected as distances grow compared to the scattering length.

While Tensor of Inertia is used to find the characteristics of the shape of the light distribution, CFirst can also estimate the vertex time of the event using the time residual parameter. The time residual information can be used to give an estimate for the vertex time t_v of an event from the relation

$$t_{res} = t_i - \left(t_v - \frac{d_i}{c_{ice}} \right), \tag{4.3}$$

where d_i is the i^{th} hit at a distance d_i from the vertex, at a time t_i. From the estimates taken for the vertex position and time for the event, a seed is given to the likelihood maximizers mentioned in the next section.

4.3.2 Likelihood Maximization

Once the first guess methods have been performed, the *likelihood reconstructions* refine the estimates of the vertex and time of the event. For cascades, the first guess methods look at the center of gravity of the light distribution. The likelihood maximizers are designed to account for the scattering and absorption of the light, and how these factors affect the distribution of photon arrival times at each DOM.

In general, the maximizers follow the relation:

$$\mathcal{L} = \prod_{i=0}^{hits} p(t^i_{res}, d_i). \tag{4.4}$$

This likelihood function evaluates the probability of a hit occurring with a time residual t_{res}, at a distance d from the centre, for all hits in the event. The maximization method can be used for cascades and muons. For the muon, the light origin is constantly changing as the charged lepton moves throughout the detector, and therefore the maximizer uses coordinates for a moving frame of reference. For cascades, the light origin is approximated to be stationary at the vertex. The probability $p(t^i_{res}, d_i)$ in equation 4.4 is a parametric function evaluated from the analysis of Monte Carlo data based on light distributions in the ice. This probability distribution is also known as the *Pandel Function* [25], and is expressed as a function of τ, λ, λ_a, c_{ice} - the scattering time, the absorption and scattering lengths, and the speed of light within the ice respectively:

$$p(t, d) = \frac{\tau^{(-d/\lambda)} t^{(d/\lambda - 1)} e^{-(t/\tau + c_{ice} t/\lambda_a + d/\lambda_a)}}{\Gamma(d/\lambda)}. \tag{4.5}$$

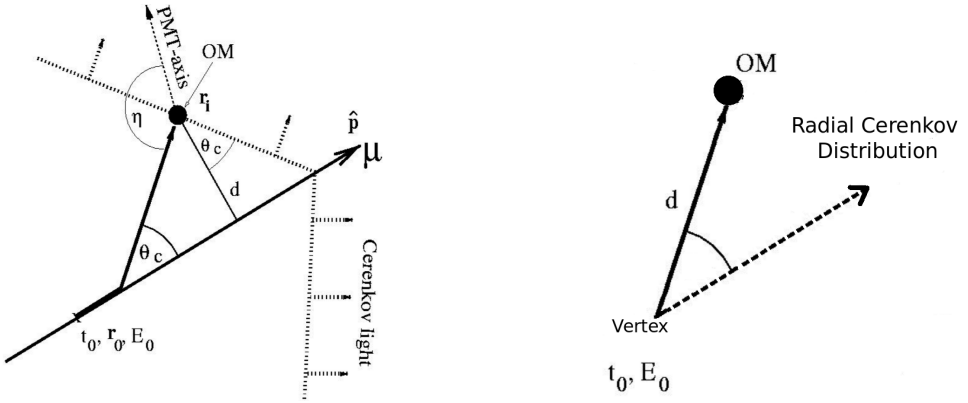

Figure 4.2: Cerenkov coordinate systems for a muon track, and a cascade event. For the muon track, the light origin is at the moving frame of reference. The cascade produces all of its Cerenkov light at the vertex, which is diffusely scattered by the ice.

First guess methods use the data from the first 'hit' on the DOM – where the times for single photoelectrons (SPE) are recorded. Because a cascade can produce large amounts of light visible to the detector, a single DOM close to the vertex could receive many hundreds of photons for a single event. Because of this, using a distribution based on the timings of the first detected photoelectron is inadequate. A modified probability density function exists for the case of multiple photoelectrons (MPE). It is described as the probability of finding N photons at a distance d with a time after that of the first time residual $t_{res} = t$:

$$p(N, t, d) = Np(t, d)\left(\int_t^\infty dt' p(t, d)\right)^{(N-1)}. \tag{4.6}$$

Since the DOMs are susceptible to electronic noise, a time jitter factor must be added to equation 4.5. This factor removes the divergence of t at zero.

Cscdllh

Cscd-llh, or Cascade Likelihood, is the main cascade based reconstruction module used to estimate the positions, times and energies of cascade events. Cascade Likelihood uses different probability density functions (PDFs) for maximizing on the vertex parameters (*UPandel* and *UPandelMpe*), and energy parameters (*HitNoHit* and *HitNoHitMpe*) [26]. The PDF *UPandel* is also known as a Convoluted Pandel function. *UPandel* accounts for the time jitter mentioned earlier, by describing the overall distribution with three functions which operate over a time interval around a

patch time $t_1 = \sqrt{2\pi}\sigma$, where σ is the Gaussian width.

$$P_U(d,t) = \begin{cases} P_1 & \text{for } t < 0 \\ P_2 & \text{for } 0 < t < t_1 \\ P_3 & \text{for } t > t_1. \end{cases} \tag{4.7}$$

In this expression, the piecewise functions P_1, P_2, P_3 are Gaussian, 3rd order polynomial and unpatched Pandel functions respectively:

$$P_1 = \frac{1}{\sqrt{2\pi}\sigma} e^{(-t^2/2\sigma^2)} \tag{4.8}$$

$$P_2 = c_0 + c_1 t + c_2 t^2 + c_3 t^3 \tag{4.9}$$

$$P_3 = p(t,d). \tag{4.10}$$

With c_0 : c_3 being the polynomial constants each of a certain value so that the piecewise function P_U is continuous and differentiable at the limits given by equation 4.7. These constants are found via Monte-Carlo studies on the light arrival times at each DOM, in order to fit the Gaussian distribution with the Pandel function. The

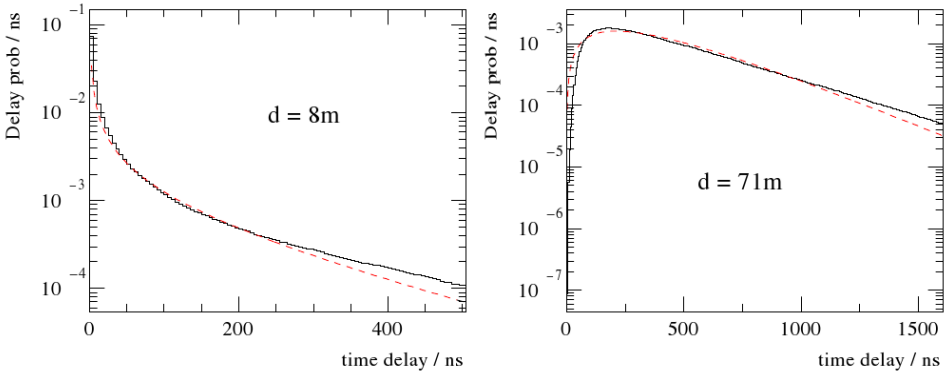

Figure 4.3: Time residuals for the Patched Pandel function (equation 4.7)at two distances from a Cereknov source (dashed line). Solid line represents time residuals obtained from simulated Monte Carlo data

results from the UPandel PDF are then seeded to the UPandelMpe routine. The UPandelMpe PDF can be described by the generalised multi-photoelectron function equation 4.6, with an added constant p_0, used to account for uncorrelated noise hits.

$$p(N,t,d) = N[P_U(t,d) + p_0]\left(\int_t^\infty dt' p(t,d) + p_0\right)^{(N-1)} \tag{4.11}$$

The energy routines *HitNoHit* and *HitNoHitMpe* are used to obtain estimates for the energy based on the relative probability of a hit occurring at a DOM to the probability of the DOM not being hit. Since light at distances from the vertex larger than the scattering length is diffusely scattered, the number of photons a DOM receives at this distance from the cascade is never constant. A Poisson distribution can be used to express the probability of a DOM receiving N photons given the light distribution parameter μ;

$$P(N) = \frac{\mu^N \cdot e^{-\mu}}{N!}. \tag{4.12}$$

The parameter μ itself is related to the cascade energy, E via

$$\mu \approx \frac{I_0 E}{d_{thres} + d} e^{(-d/\lambda_a)}, \tag{4.13}$$

where I_0 is an intensity constant derived from experimental Monte Carlo data, λ_a is the attenuation of light in the medium and d_{thres} is a constant related to the scattering length. The constant d_{thres} limits the equation to a range greater than this length, where the light is diffusely scattered ($I_0 \sim 1/d$), and not radially scattered ($I_0 \sim 1/d^2$).

The HitNoHit PDF evaluates the probability of either a photon from the cascade or a noise hit in the PMT producing a single photoelectron at a given DOM. This probability is expressed as a function of the individual probabilities P_{noise} and P_{hit}^{casc}. For a single photoelectron, the probability of a hit on a DOM caused by light from the cascade is $P_{hit}^{casc} = 1 - e^{-\mu}$, while the probability of a hit not occurring is $P_{nohit}^{casc} = e^{-\mu}$. The total probabilities for each case are then

$$P_{hit} = (1 - P_{dead}) P_{hit}^{casc} + P_{noise} P_{nohit}^{casc}. \tag{4.14}$$

This relation adds the individual probabilities of a hit from the cascade, the probability of a hit from noise, minus the probability of a cascade photon hitting a dead DOM. Likewise, the overall probability of a hit not occurring is expressed as

$$P_{nohit} = P_{dead} P_{hit}^{casc} + (1 - P_{noise}) P_{nohit}^{casc}. \tag{4.15}$$

Here P_{dead} is the probability of a DOM not responding to any light signal, regardless of intensity. The likelihood function for the HitNoHit PDF is then expressed as the product of these two probabilities over all DOMs in the detector array

$$\mathcal{L} = \prod_{i=0}^{\text{allhits}} P_{\text{hit}(E,d)} \prod_{j=0}^{\text{unhit}} P_{\text{nohit}(E,d)}. \tag{4.16}$$

HitNoHitMpe uses the Poisson distribution for N photoelectrons along with an energy seed from the previous module for a multi-photoelectron estimate on the energy of the cascade. The likelihood function in this case is

$$\mathcal{L} = \prod_{i=0}^{\text{allDOMs}} P(N; E, d), \tag{4.17}$$

where

$$P(N; E, d) = \begin{cases} P_{casc}(0)(1 - P_{noise}) & \text{for } N = 0 \\ P_{casc}(N)(1 - P_{noise}) + P_{casc}(N-1)P_{noise} & \text{for } N > 0 \end{cases}, \tag{4.18}$$

and

$$P_{casc}(N) = \frac{\mu^N \cdot e^{-\mu}}{N!}. \tag{4.19}$$

Equation 4.18 describes the probability of a noise hit occurring on a DOM at $N = 0$, where no photons reach the PMT, and the probability of N hits caused by N photons using the Poisson distribution from equation 4.19.

AtmCscdEnergyReco

AtmCscdEnergyReco is a modification of an older module *CVertex*, with emphasis on a cleaner and easier photonics based implementation. This module uses specialised photonics reconstruction or *photorec* tables in order to account for the difference in light distributions caused by absorption and scattering as a function of depth in the detector [1]. This energy reconstruction method uses equation 4.19 to solve for $\frac{d\mu}{dE}$ in order to find the energy of the cascade as a function of the number of detected photoelectrons

[1] Details on the creation of the flasher tables used in the analysis in the following section can be found at *http://wiki.icecube.wisc.edu/index.php/Flasher_Simulation*

$$E[GeV] = \frac{\sum N}{\sum \frac{d\mu}{dE}(1\ GeV)}. \tag{4.20}$$

Here $\sum \frac{d\mu}{dE}$ is the amplitude of a 1 GeV cascade for a given vertex position in the detector, based on the photorec tables that were supplied to the AtmCscdEnergyReco module. The amplitude for a 1 GeV cascade will vary greatly over the detector, due to the depth dependence of scattering and the absorption. *AtmCscdEnergyReco* can also take into account the noise hits on each PMT as with the *Cscd-llh:HitNoHit* routine. Because of the large differences in light distributions as a function of depth for events centered around locations such as dust layers, a good vertex seed is crucial for any photorec based module. When using photorec tables based on flasher events, *AtmCscdEnergyReco* is particularly sensitive, as the tables are created by the simulated propagation of light unique to the dust and gas bubbles around the center of the flashing DOM. If the vertex estimate is off by a few meters, the module will load a photorec table representing the hit optical modules from the next flashing DOM on the string, where the activated DOMs for that particular light distribution are slightly different.

Rime

Rime is a self contained cascade reconstruction module [27]. *Rime* obtains a vertex position estimate for an event, and then uses the seeded information to calculate the energy of an event. This makes *Rime* similar to the *AtmCscdEnergyReco* module, but *Rime* differs by not having to rely on pre-generated photorec tables. The energy estimate can be obtained from

$$-\log P\left(\{n_i\}|\{\mu_i\}\right) = -\sum_{i=1}^{k} n_i \log(\mu_i/\mu) + \sum_{i=1}^{k} n_i \log(n_i!) - N\log\mu + \mu, \tag{4.21}$$

where $n_i\log(\mu_i/\mu)$ is the hit position/timing likelihood, and μ is the estimated energy density parameter [28]. The energy estimate μ, obtained using equation 4.13 is constructed according to the differing scattering and absorption properties of ice as a function of depth in the detector. The average propagation length can be found via

$$\frac{1}{\langle \lambda_{abs,es} \rangle} = \frac{1}{R_f - R_i} \int_{R_i}^{R_f} \frac{d\,R}{\lambda_{abs,es}(R)}. \tag{4.22}$$

Because *Rime* has to look at the ice properties surrounding the seeded vertex position value for each event, this makes the reconstruction method very slow to give an estimate for the energy.

5 Results and Analysis

This chapter details the flasher runs taken at the South Pole and used in this analysis, and the evaluation of the reconstruction algorithms to accurately estimate the vertex parameters of the event, such as the position of the light origin and the number of photons produced by the flashing LEDs. The second section covers the analysis done with two different flasher brightness settings with full string readout in the detector, and describes the problems encountered when reconstructing the vertex position and estimated number of photons for some of the modules. The third section describes an identical analysis taken with the whole string containing the flashing DOM turned off from readout. This was done in the hope of fixing the shortfalls observed for some of the reconstruction modules.

5.1 Flasher Runs 111739-44

The flasher runs 111739–11144 were taken in September 2008 after some initial problems with the DAQ configuration. These problems were due to the large amount of light generated by the flashers, which saturated the input/output capability of the detector software to record all of the information. To reduce the problems while taking the data, the flashers were run at a low flashing frequency to reduce the load on the DAQ system. This reduced the number of overall events from what was originally planned. The data was pre-processed at the pole, and all of the flasher events were selected and separated from the muon background. The flasher runs were then uploaded via satelite to the IceCube data warehouse in Madison, Wisconsin. Details on the runs are shown in table 5.1.

The flasher data was then split into events according to the flasher position, which made analysis easier. Monte Carlo data was also generated to match the statistics and input parameters for each flashing DOM. The next two sections will cover the analysis of the data with the full detector readout, and the data analysed with the flashing string ignored from the readout.

Run - Subrun Number	Flashing DOMs	Relative LED Brightness (Hex Mask)
111739 – 00000001	05, 10, 15, 20, 25, 30, 35, 40, 45, 50, 55, 60	127
111739 - 00000003	05, 10, 15, 20, 25, 30, 35, 40, 45, 50, 55, 60	63*
111739 - 00000005	04, 09, 14, 19, 24, 29, 34, 39, 44, 49, 54, 59	127
111739 - 00000007	04, 09, 14, 19, 24, 29, 34, 39, 44, 49, 54, 59	63
111739 - 00000009	03, 08, 13, 18, 23, 28, 33, 38, 43, 48, 53, 58	127
111740 - 00000001	03, 08, 13, 18, 23, 28, 33, 38, 43, 48, 53, 58	63
111741 - 00000001	02, 07, 12, 17, 22, 27, 32, 37, 42, 47, 52, 57	127
111744 - 00000003	02, 07, 12, 17, 22, 27, 32, 37, 42, 47, 52, 57	63
111741 - 00000005	01, 06, 11, 16, 21, 26, 31, 36, 41, 46, 51, 56	127*
111744 - 00000001	01, 06, 11, 16, 21, 26, 31, 36, 41, 46, 51, 56	63

Table 5.1: Detailed information for the flasher runs 111739-111744. * Indicates the data for the particular subrun is corrupted, or unavailable. Each subrun consists of roughly 286 events per flashing DOM. Each DOM had a pulse width of 62ns, corresponding to 1.30 $\times 10^{10}$ and 6.51$\times 10^{9}$ produced photons [29] for the full and half brightness runs respectively.

5.2 Flasher Reconstructions - String 63 readout enabled

The first half of the analysis on the flasher runs 111739–111744 covers all of the reconstructions performed on data taken from all available strings in the detector. Using the data from all strings is generally important, as it maximizes the amount of data the software is able to look at for a particular event. However, in the case of high energy flasher events, problems with the close DOM spacing on the flashing string can cause large errors in the reconstructed vertex z position, which is described in the next subsection. Because of these vertex problems, a second analysis was done with the flashing string disabled from readout, and the results are covered in the next section. For the following energy and position reconstructions of the vertex, the reconstruction seeding process is shown schematically in figure 5.1, and is described in detail in the appendix.

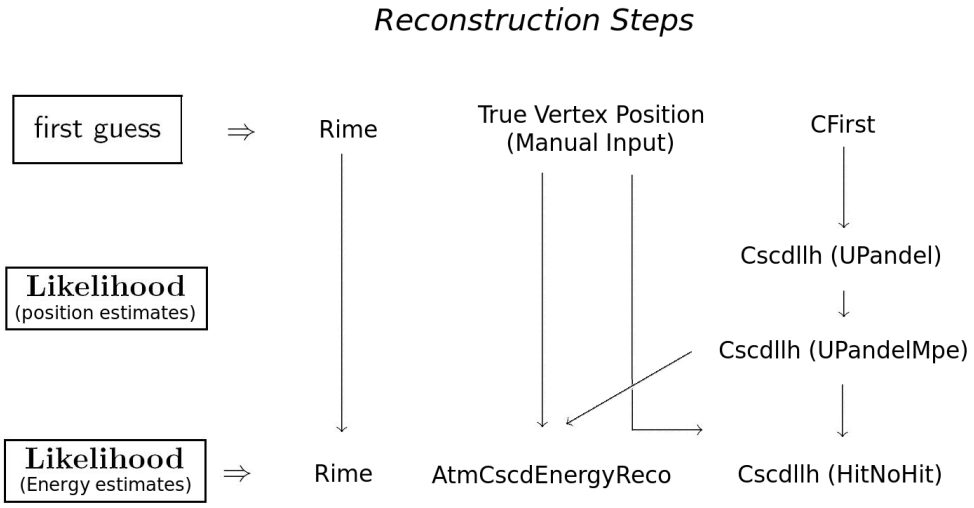

Figure 5.1: Schematic process of the seeding chain used by the modules in the analysis of the flasher runs 111739–111744

5.2.1 Vertex Position Reconstructions

All of the energy estimates depend on a seed for the vertex position of the cascade. A major cause in the distribution of energies that are reconstructed is often due to minor variances in the estimated vertex position that the energy reconstruction module was seeded with. Because the energy reconstruction modules display this sensitivity, it is important to understand the variance in reconstructed positions as a function of depth. Figures 5.2 - 5.5 show the reconstructed position estimates for the vertex from the 4 different position reconstruction methods used in the analysis, with the root mean square (RMS) of the distributions shown as the error bars. The vertex estimates described in this section are shown from one brightness setting only, as there was very little variation between the position estimates obtained from the two different brightness settings.

The *CFirst* method gave position estimates for the vertex consistent with a center of gravity approach; where the z vertex estimate lies above the true position for events above a dust layer (due to the shaping of the overall light distribution from the absorbing effects of the dust layer), and vice versa for events below a dust layer. This is in accordance with the dust model shown in figure 2.5, which shows the scattering and absorption of light as a function of wavelength and depth in the detector. *CFirst* had a relatively low RMS for all of the 3 vertex estimates, which is due to the fact that the module was not processing weighted hits: only the first hits from each DOM were used to calculate the vertex, and so the results are fairly consistent with one another.

The *(Cscd-llh) UPandel* method had a much larger RMS for all of the position estimates, as shown in figure 5.3. This is due to the weighting setting used by the module, which processed all hits on each DOM, where the number of hits varies more than the distribution in timing of the first hit on all of the DOMs that observed the event. This fact led *UPandel* to perform noticeably better in the z estimate than *CFirst* however, because of the large number of hits on DOMs near the flasher, which pulled the overall estimate to a value near the true position.

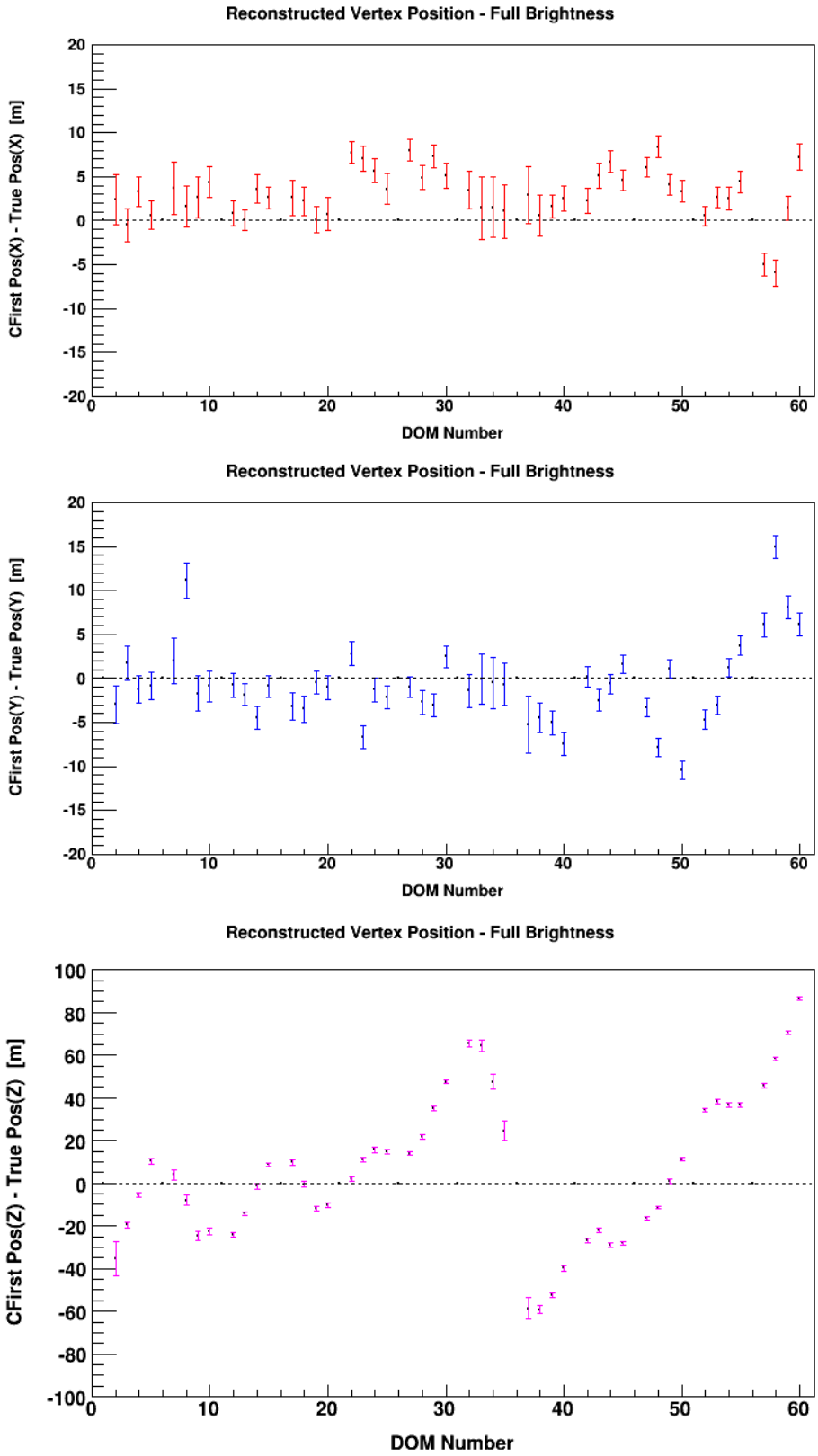

Figure 5.2: CFirst x, y, z vertex position reconstructions for the flasher runs at full LED brightness, where the dotted line shows the position of the true value.

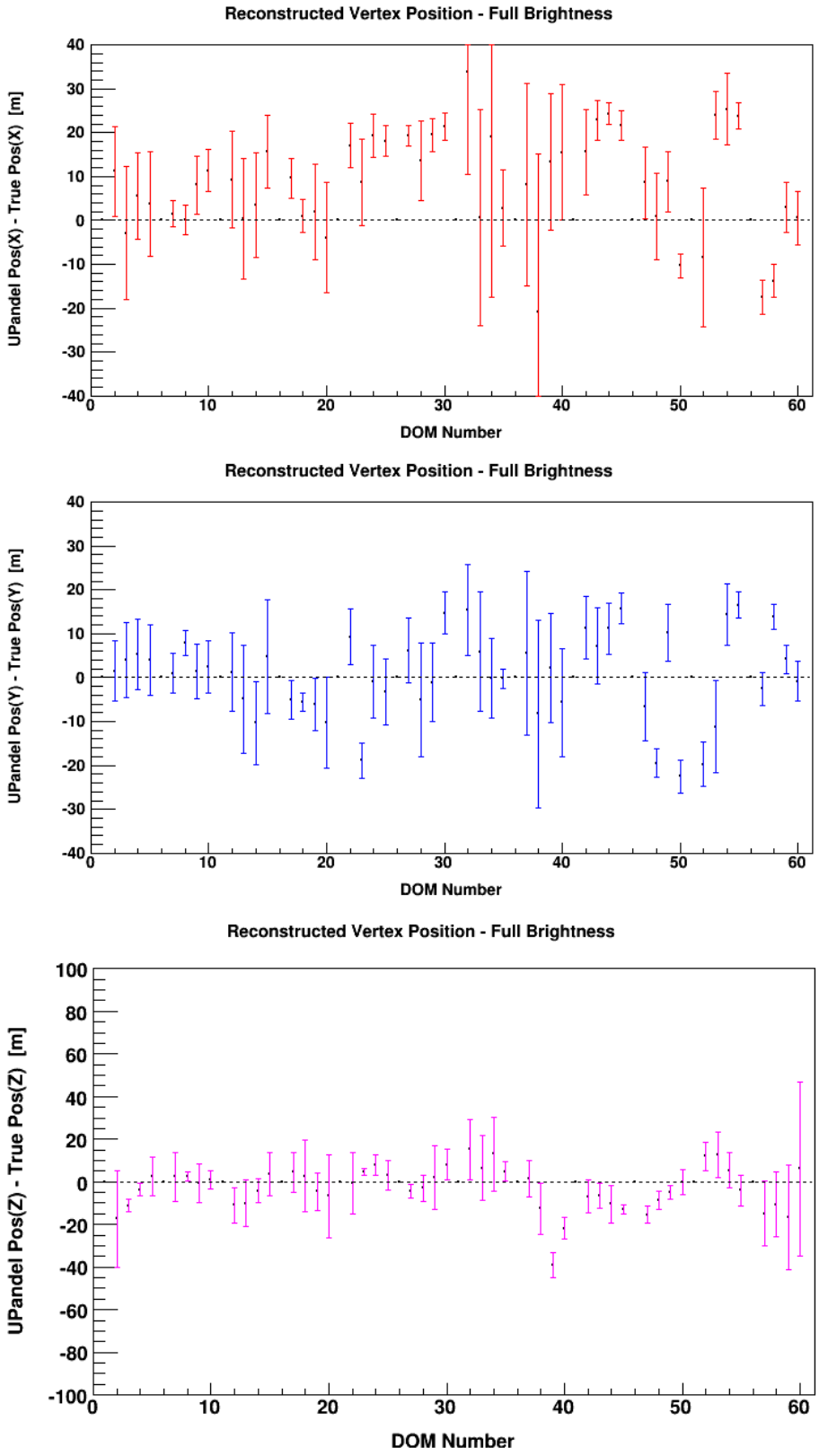

Figure 5.3: (Cscd-llh) UPandel x, y, z vertex position reconstructions for the flasher runs at full LED brightness, where the dotted line shows the position of the true value.

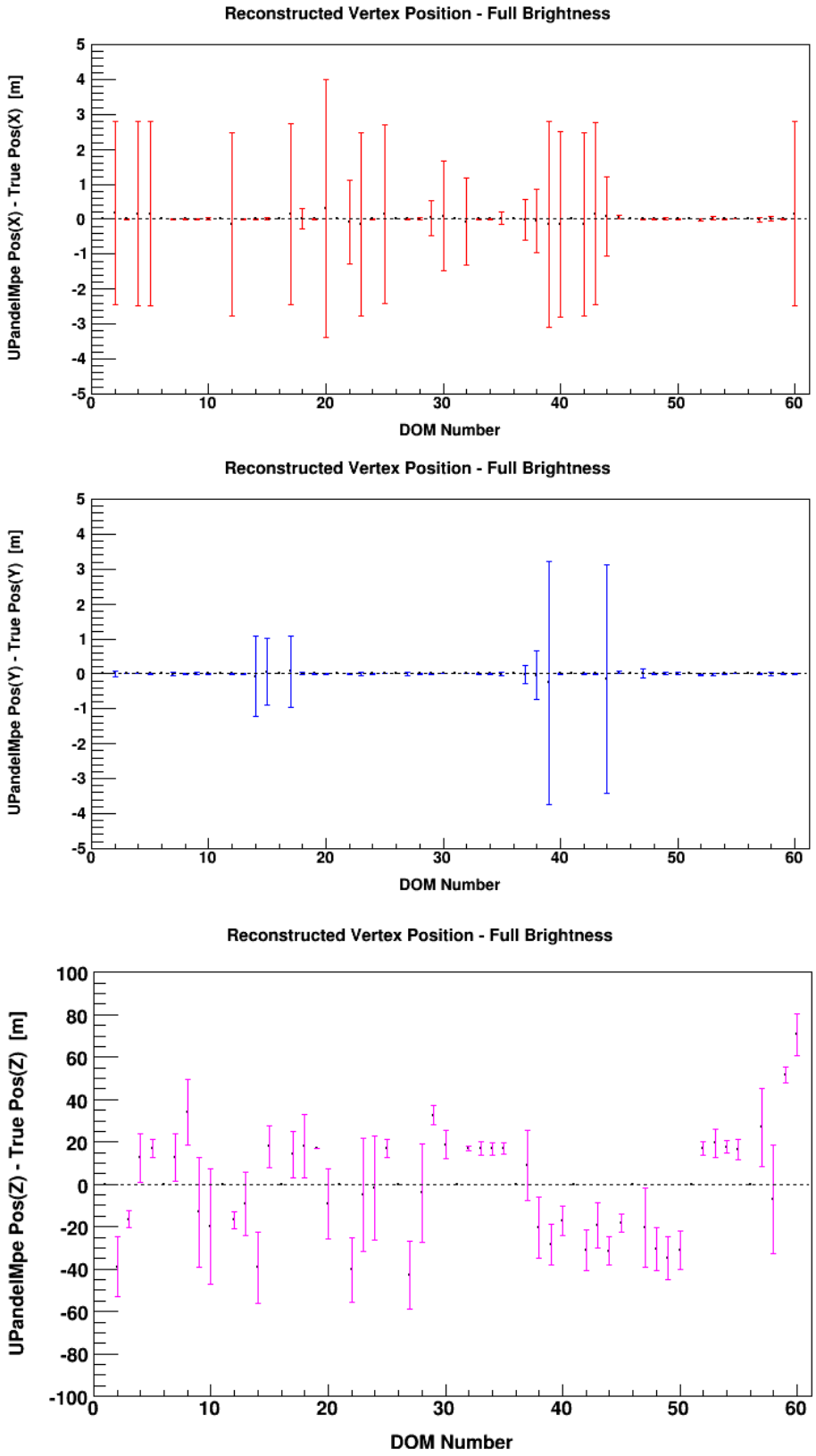

Figure 5.4: (Cscd-llh) UPandelMpe x, y, z vertex position reconstructions for the flasher runs at full LED brightness, where the dotted line shows the position of the true value.

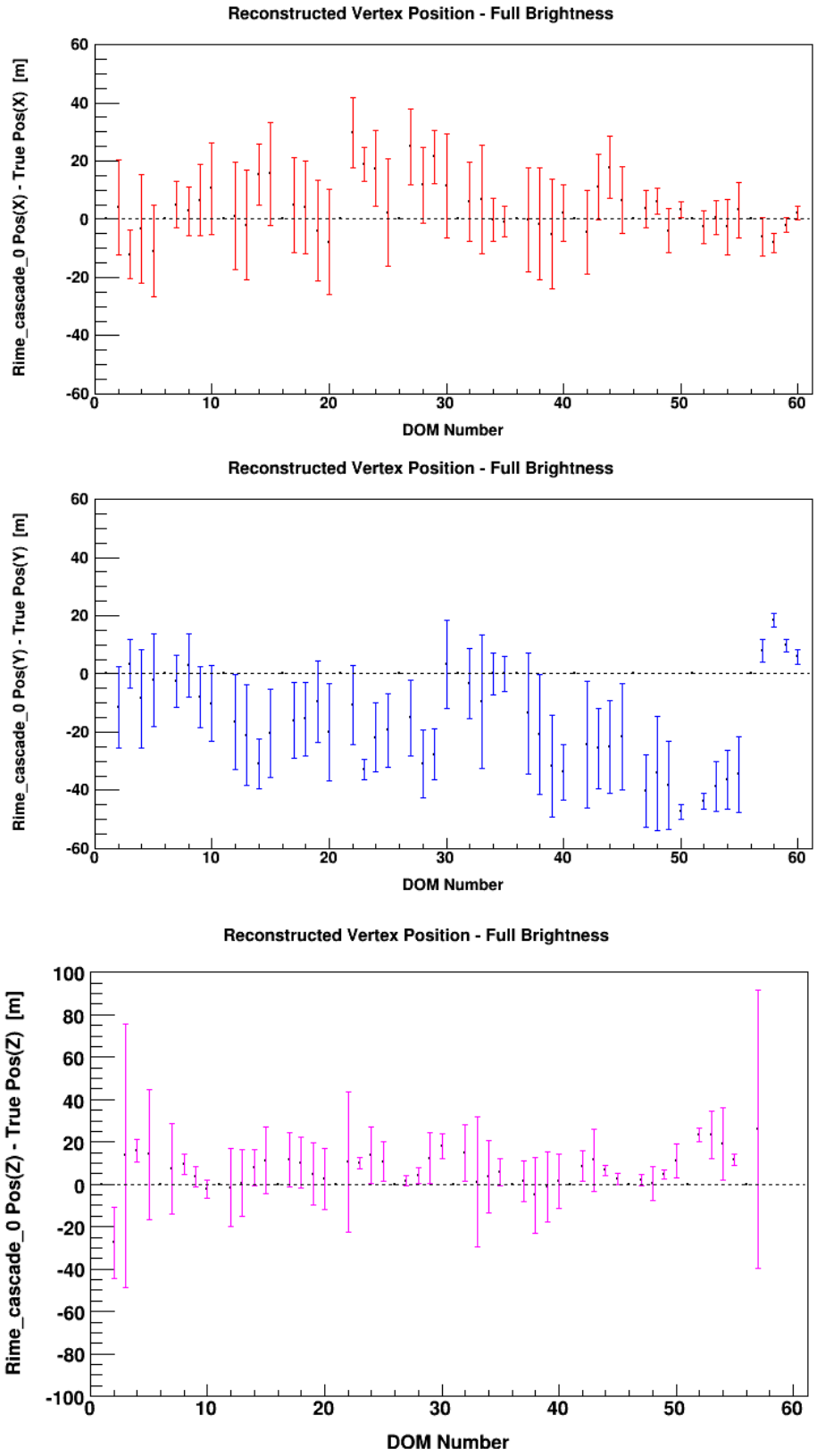

Figure 5.5: (Cscd-llh) Rime x, y, z vertex position reconstructions for the flasher runs at full LED brightness, where the dotted line shows the position of the true value.

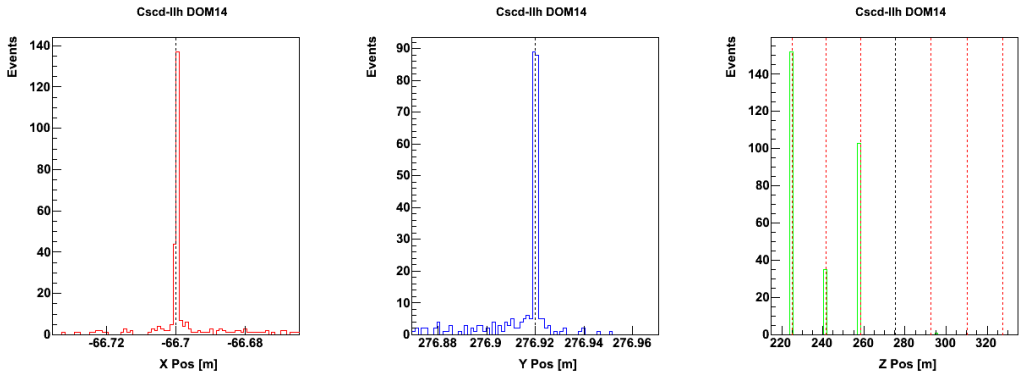

Figure 5.6: (Cscd-llh) UPandelMpe x, y, z vertex position reconstructions for the flasher events centered at DOM14. The red dashed lines in the third graph represent the positions of the nearest DOMs on the flashing string. The black dashed line represents the flasher origin (DOM14). Of interest is the large accuracy for the position estimates in x and y, and the splitting of estimates in the reconstructed z position.

The results for the *(Cscd-llh) UPandelMpe* position estimates in figure 5.3 show a very accurate result in x and y, with very small RMS values for some of the flashing DOMs. The *UPandelMpe* routine displays a tendency to always reconstruct the vertex origin on a DOM position. This is again due to the large number of hits on DOMs nearest the flashers, but the result is even more compressed by the UPandelMpe PDF, which maximizes on the arrival probabilities of each photon in a hit DOM. *(Cscd-llh) UPandelMpe* performs poorly in the z estimate as a side effect however, which can be explained by the close spacing of the reading DOMs on the string, with respect to the spacing between strings in the XY plane.

The close spacing of the DOMs on the string is also the cause for the large RMS values for some of the flashing DOMs in the z position ($\sim$ 20m). For these events, the reconstructed z position for a single flashing DOM is split over multiple values, all of which correspond to DOMs nearest the flasher. An example of this is illustrated in figure 5.6.

The vertex problem can be described by the effects of the large numbers of photons detected by the DOMs nearest the vertex. Equation 4.6 describes the probability of receiving N photons at a distance d each at a time t greater than the time of the first time residual t_{res}. For large values of N, the probability $p(N, t, d)$ has an increasingly sharp peak at $t_{res} = 0$. Then this DOM starts to dominate the entire PDF for all DOMs. This causes the *UPandelMpe* routine to be very sensitive to the vertex estimate it is seeded with. Since the flashing DOM is automatically excluded from

readout, the vertex estimates from *UPandelMpe* are pulled away from the true position; to DOMs in the range of the seeded vertex values, that receive large amounts of light. If the charge seen by DOMs on the flashing string near the origin of the event receive a similar amount of light with respect to one another, a widely distributed seed for the vertex estimate can cause the splitting of z positions seen in figure 5.6. This concludes that the *UPandelMpe* PDF has difficulties if the true vertex is close (< 20m) to the DOMs. This effect occurred for both of the LED brightness flasher runs.

5.2.2 Full Brightness Energy/Nph Estimates

The full brightness runs are described in table 5.1. Overall, the results from the reconstruction analysis differ little from the analysis with the half brightness runs, because of the small difference in the number of photons produced by an event (10.18 vs 9.81 [$\log_{10}$ NPh] for full and half brightness LED settings respectively). Figure 5.7 displays the total charge seen by all of the DOMs for the flasher events, as a function of depth. The spread in the total received charges per flasher event puts a lower limit on the distributions of the Energy/NPh estimates from each of the reconstruction modules.

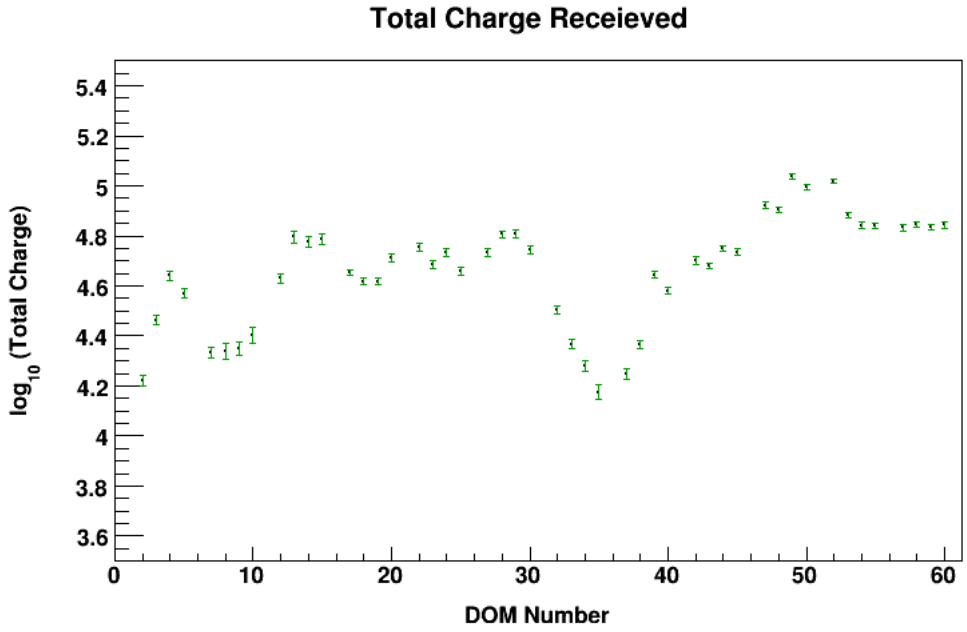

Figure 5.7: Total Charge observed by all of the hit DOMs for the flasher runs at full LED brightness.

With the exception of *Rime*, the reconstruction modules return an estimate of the energy of a cascade, which is not applicable to a flasher event. To obtain an estimate for the number of photons, a conversion factor of 1.37×10^5 NPh/GeV [footnote needed] was used [1] for the *Cscd-llh HitNoHit* module to obtain the plot in figure 5.12. *AtmCscdEnergyReco* used a photonics based implementation to estimate the energy of a cascade based on a similar conversion factor to *Cscd-llh HitNoHit* (1.39×10^5 NPh/GeV). Since this conversion is implemented in the module, an inverse conversion for the number of estimated photons was made to obtain the plots

[1] This factor: 1.37×10^5 NPh/GeV was described in a talk at the Baton Rouge collaboration Meeting, 2006 [28]

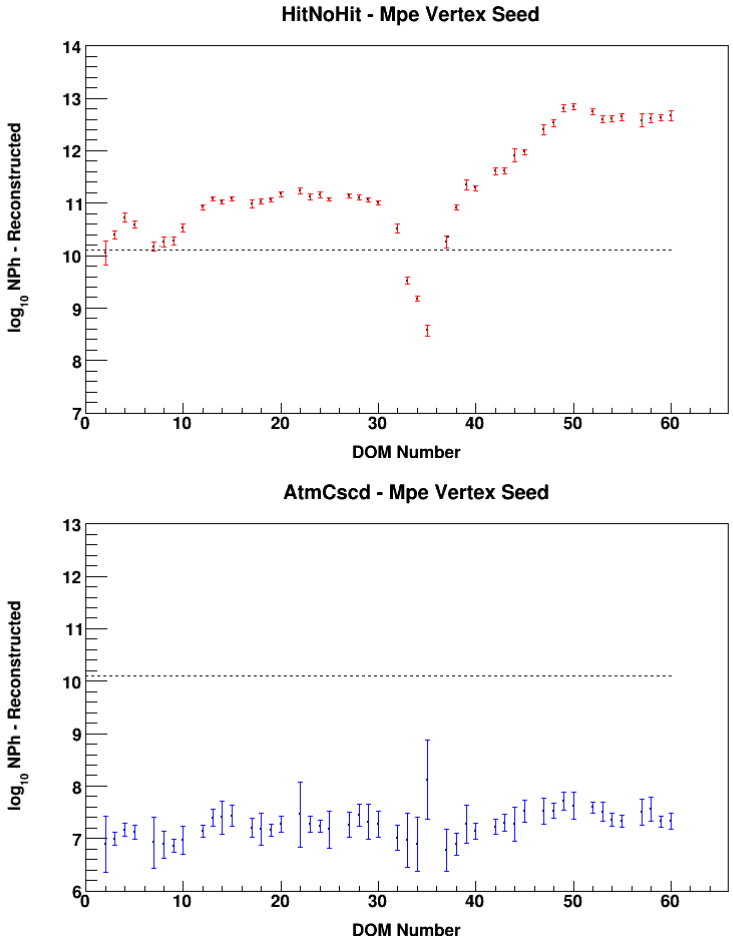

Figure 5.8: NPh (Number of Photons) estimates for each flashing DOM from the UPandelMpe vertex seed for the flasher runs at full LED brightness, where the dotted line shows the position of the true value.

in figures 5.12 and 5.13. *Rime* directly gave the number of photons via the n_0 parameter, and so no conversion factors were necessary when plotting figure 5.14.

The *Cscd-llh HitNoHit* routine shows very little sensitivity to the seeded vertex position. This is illustrated by the small difference between the first plots in figure 5.8 and in figure 5.8. The routine also displays large depth variance in the reconstructed number of photons, with the estimates in the clear ice (around DOM50), being 2-3 orders of magnitude higher than the produced number of photons. This large depth variance is expected, as the *Cscd-llh HitNoHit* routine performs the cascade energy calculations with an average of the ice properties over values obtained from the AMANDA detector, which does not cover the range of ice properties seen in IceCube.

Figure 5.12 displays the sensitivity of the *AtmCscdEnergyReco* module to the seeded position estimate, which is characteristic of using the flasher based photorec tables

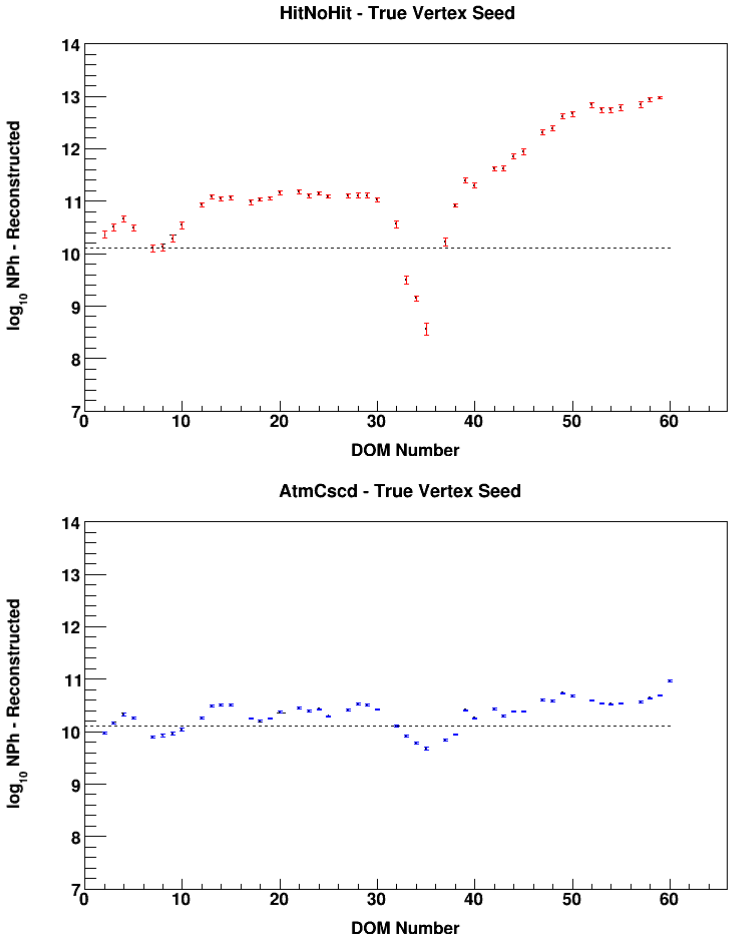

Figure 5.9: NPh (Number of Photons) estimates for each flashing DOM from the True vertex seed for the flasher runs at full LED brightness, where the dotted line shows the position of the true value.

in the reconstruction algorithm. When the vertex estimate produces a value for the position far enough from the true vertex, the photonics based reconstruction module will load a table corresponding to a different light distribution than that of the event it is looking at. This is ever more extreme for the case of bright flasher events. For the flasher runs 111739–111744, a unique photorec table was generated for each DOM. If the vertex was off by more than half the DOM spacing, the photorec table for the adjacent DOM would be loaded. The difference in the hit OM pattern between the two tables would lead to an overall low estimate for the number of photons. If *AtmCscdEnergyReco* is seeded with the true vertex position, it produces very good estimates for the number of photons produced by the LEDs, with a small spread in the estimates for each flashing DOM (figure 5.12). Spread in the reconstructed number of photons for each DOM for both *Cscd-llh HitNoHit* and *AtmCscdEnergyReco* when seeded with the true vertex position arises from the spread in distributions of the total charge seen by each DOM in the detector array between events.

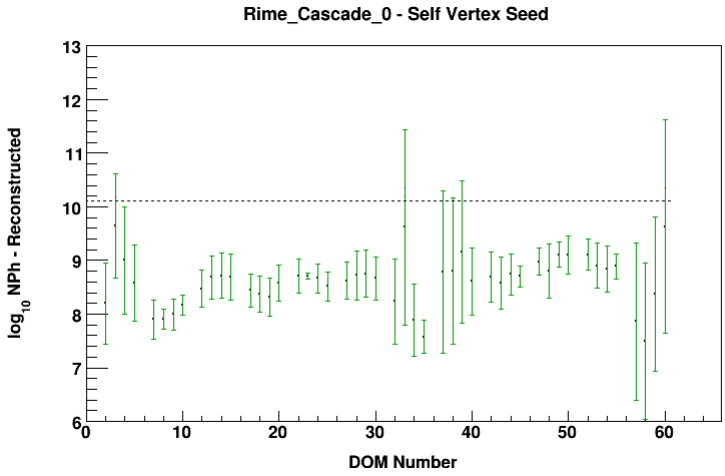

Figure 5.10: NPh estimate from the Rime energy reconstruction module for the flasher runs at full LED brightness, where the dotted line shows the position of the true value.

The number of estimated photons produced with the Rime plot in figure 5.14 are lower than the true value by two orders of magnitude. Since the *Rime* uses information on the ice properties in the detector when reconstructing, the low estimate for the number of reconstructed photons is possibly caused from a similar effect to the *AtmCscdEnergyReco* module, when seeded with an estimated vertex position. The plots showing the reconstructed vertex position used for seeding the energy estimate in Rime (figure 5.5), give evidence for vertex position sensitivity, as the estimated position for the vertex is offset by 20m on average. As a consequence of being sensitive to the z seed for the vertex position, the *Rime* estimate for the number of photons is largely affected by the big dust layer in the middle of the detector. This is shown by the large RMS values for the number of estimated photons for reconstructions around DOM35. Because the build of the *Rime* module used in the analysis does not allow for Bad/discared DOMs, the routine does not correct for the lower charge observed by the detector, and so produces a lower estimate for the reconstructed number of photons for all events.

5.2.3 Half Brightness Energy/Nph Estimates

The half brightness subruns described in table 5.1 consist of the same flashing DOMs that were flashed at full brightness in the previous run. The half brightness LED setting represents a change in the number of emitted photons (in log scale) from 10.18 to 9.81.

All of the plots produced by the three different reconstruction routines for the runs taken at a half brightness LED setting show a shift in the average estimate for the

number of photons produced by a flasher for each flashing DOM. This shift is in accordance with the 0.3 change in log scale of the LED's photon output. Figure 5.11 shows the received total charge difference with the half brightness flasher runs.

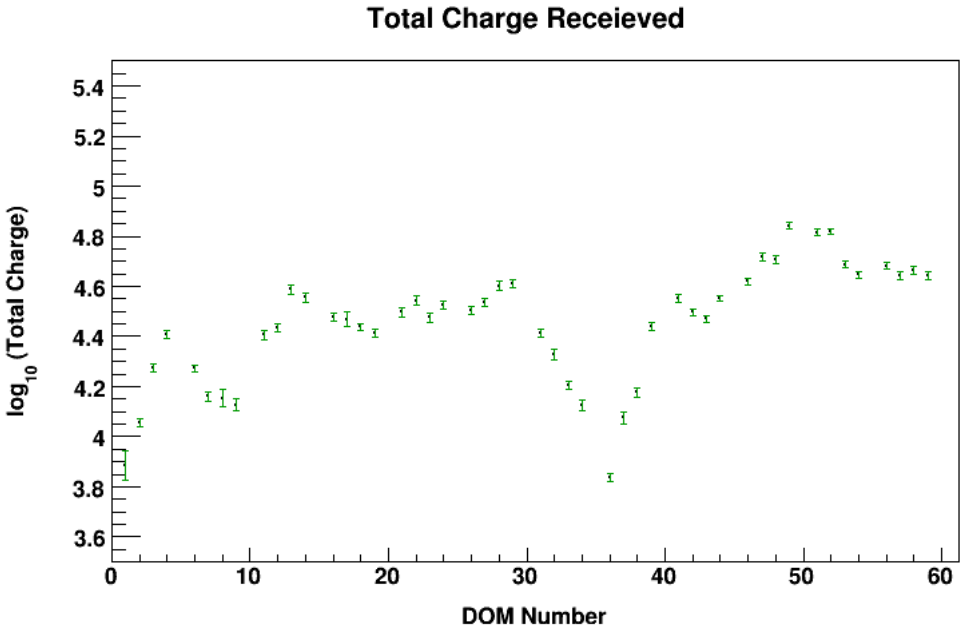

Figure 5.11: Total Charge observed by all of the hit DOMs for the flasher runs at full LED brightness.

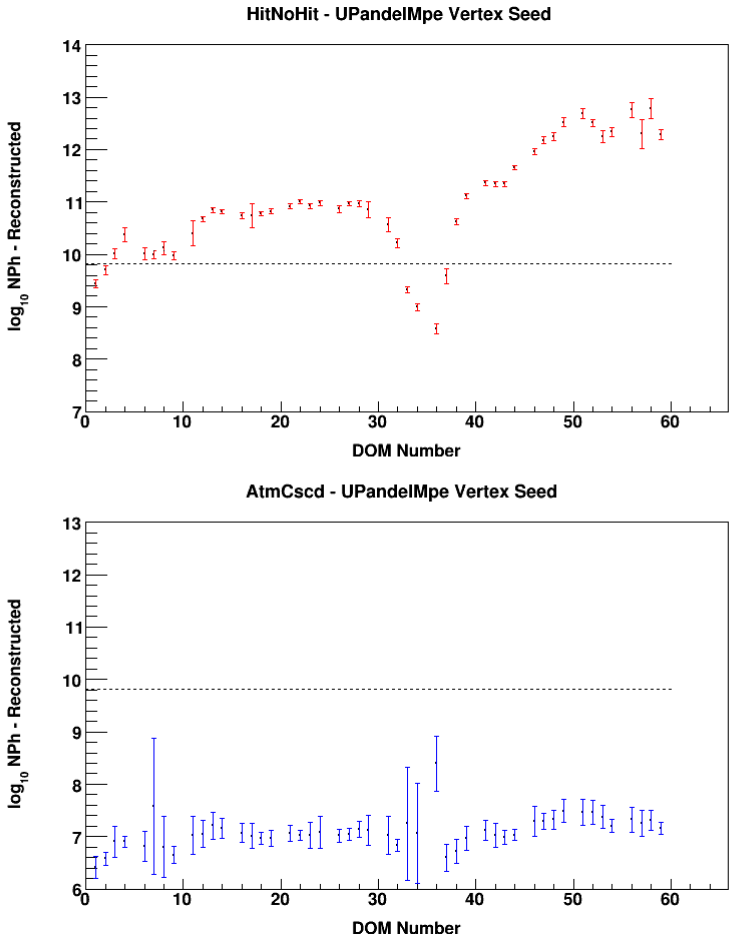

Figure 5.12: NPh estimate from the UPandelMpe vertex seed for the flasher runs at half LED brightness, where the dotted line shows the position of the true value.

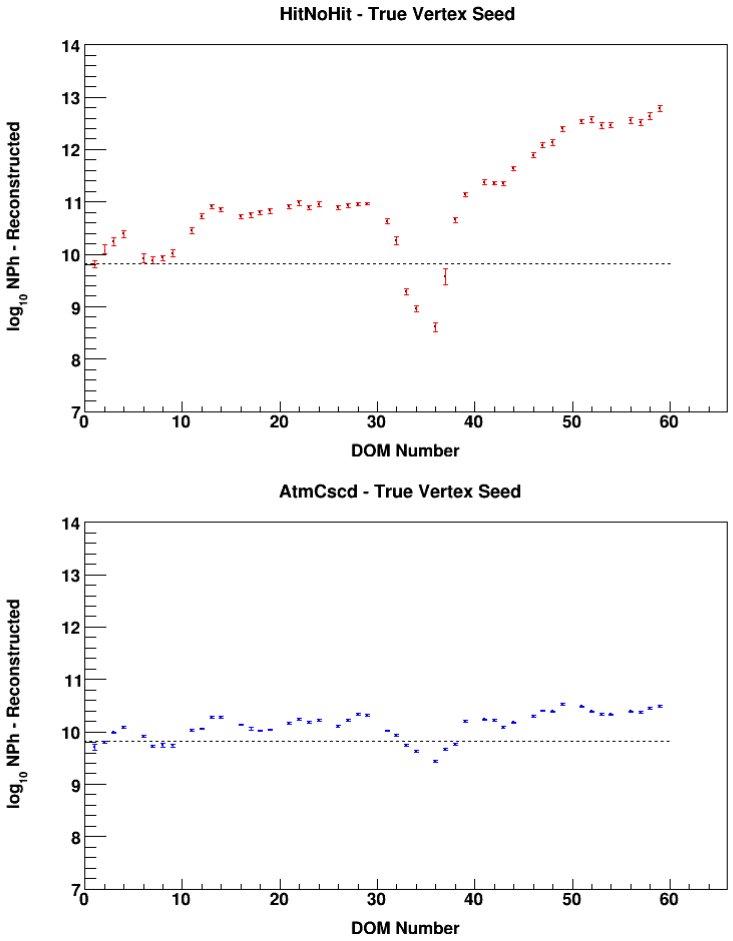

Figure 5.13: NPh estimate from the True vertex seed for the flasher runs at half LED brightness, where the dotted line shows the position of the true value.

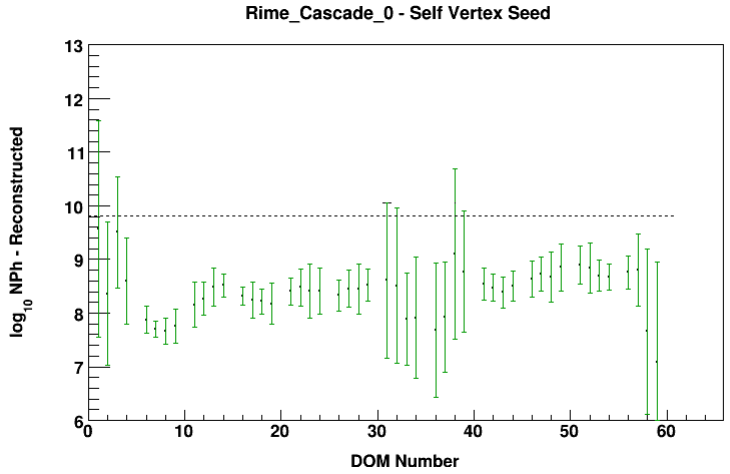

Figure 5.14: NPh estimate from the Rime energy reconstruction module for the flasher runs at half LED brightness, where the dotted line shows the position of the true value.

5.3 Reconstructions - String 63 readout disabled

Because of the biasing of the *UPandelMpe* vertex fit due to high hit counts on nearing DOMs, string 63 was excluded from the readout in the second set of analysis. This was done in order to understand how well the reconstruction modules may perform in cases where the cascade vertex lies further away from a string. Due to a limitation in the current build of the *Rime* module, the list of excluded DOMs could not be given to the reconstruction routine, which caused much lower estimates for the number of produced photons as a consequence. Because of this, *Rime* was excluded from this section of the analysis, and will be a case for future study.

5.3.1 Vertex Position Reconstructions

In order to compare the effects of disabling string 63 from readout with the results from the previous analysis, the following plots and reconstructions were generated using identical settings in the steering files.

The *CFirst* estimates for the vertex position shown in figure 5.3.1 display a wider distribution for the single DOM position reconstructions, with vertex estimates in the large dust layer having the poorest accuracy, and widest distribution. This widening of the estimates arises from the DOMs on the nearby strings having a lower consistency in the number of received photons, with respect to the DOMs nearest the light origin. Since these nearby DOMs on string 63 are ignored in the readout; with *AmpWeightPower* set to 0, the weighting is taken from the number of hit DOMs on the nearby strings, which can be largely affected by the event to event scattering of light in the ice.

UPandel improves considerably in the x and y estimates for the vertex position, as the routine parameters have an *AmpWeightPower* set to 1, which looks at the more consistent number of total hits on all DOMs surrounding the flashing string. *UPandel* is then able to refine on the seed provided by the *CFirst* routine, for larger accuracy in the XY plane. The z estimate for *UPandel* shows a larger depth dependence than was shown previously in figure 5.3.1 however, which is due to the reduced consistency in the total amount of recorded light in the z axis.

The *UPandelMpe* routine improves very little on the estimates seeded by *UPandel*, but does not produce the splitting behaviour as seen in figure 5.3.1. Without the large number of hits on the DOMs nearest the vertex, the *UPandelMpe* PDF no longer provides peaks of high probability at those DOM locations on string 63.

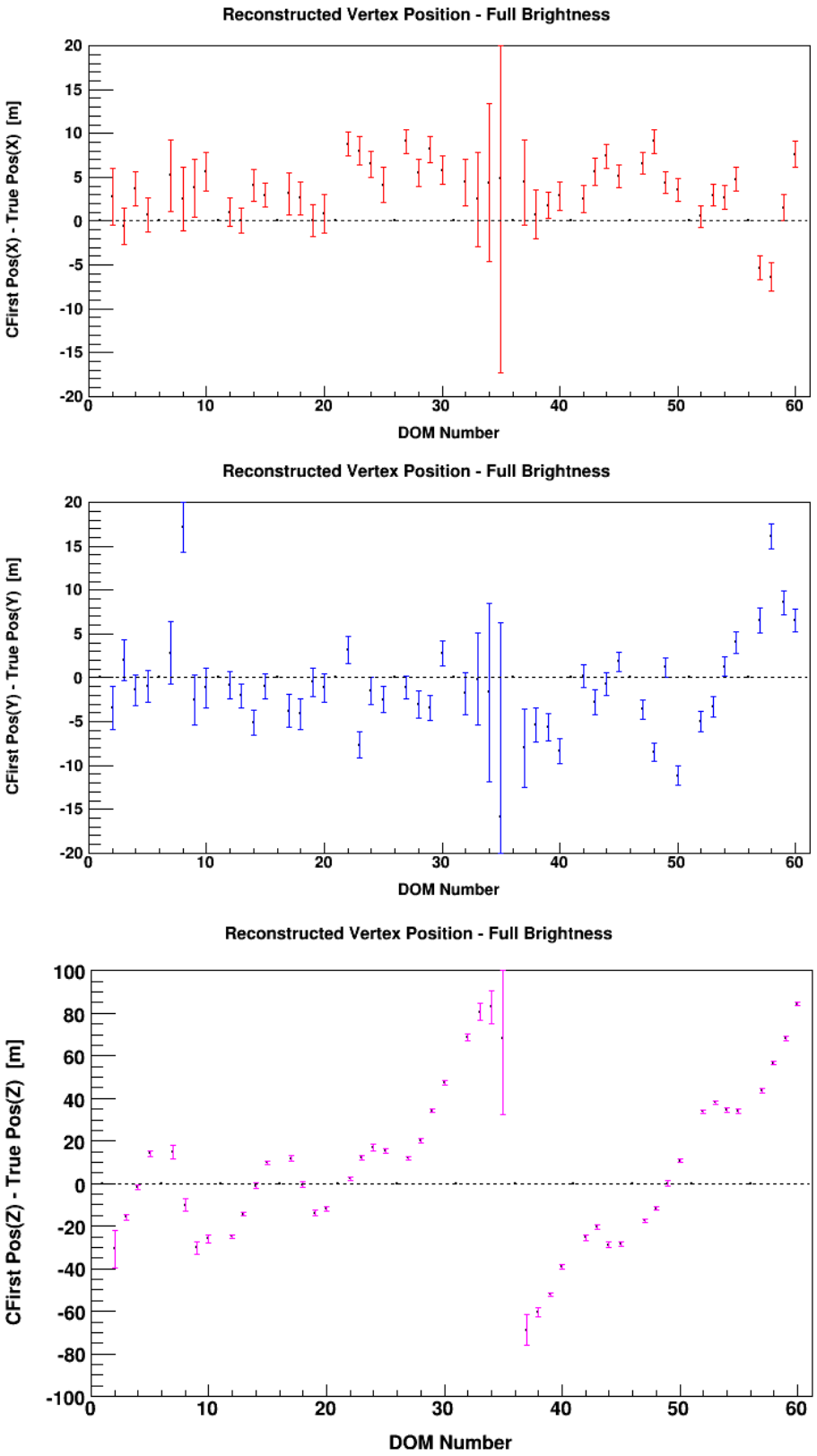

Figure 5.15: CFirst x, y, z vertex position reconstructions for the flasher runs at full LED brightness, with String 63 disabled from readout

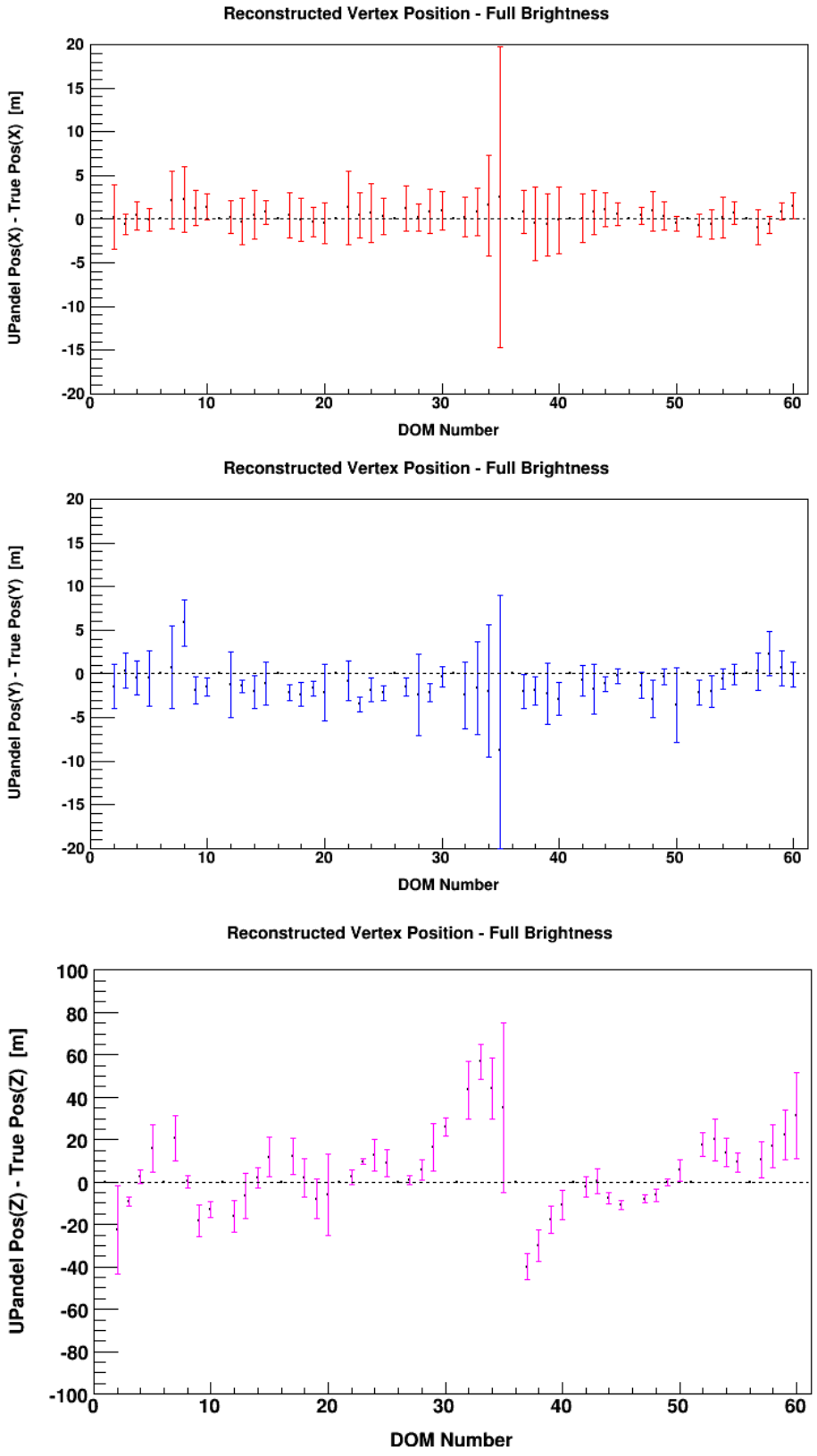

Figure 5.16: (Cscd-llh) UPandel x, y, z vertex position reconstructions for the flasher runs at full LED brightness, with String 63 disabled from readout

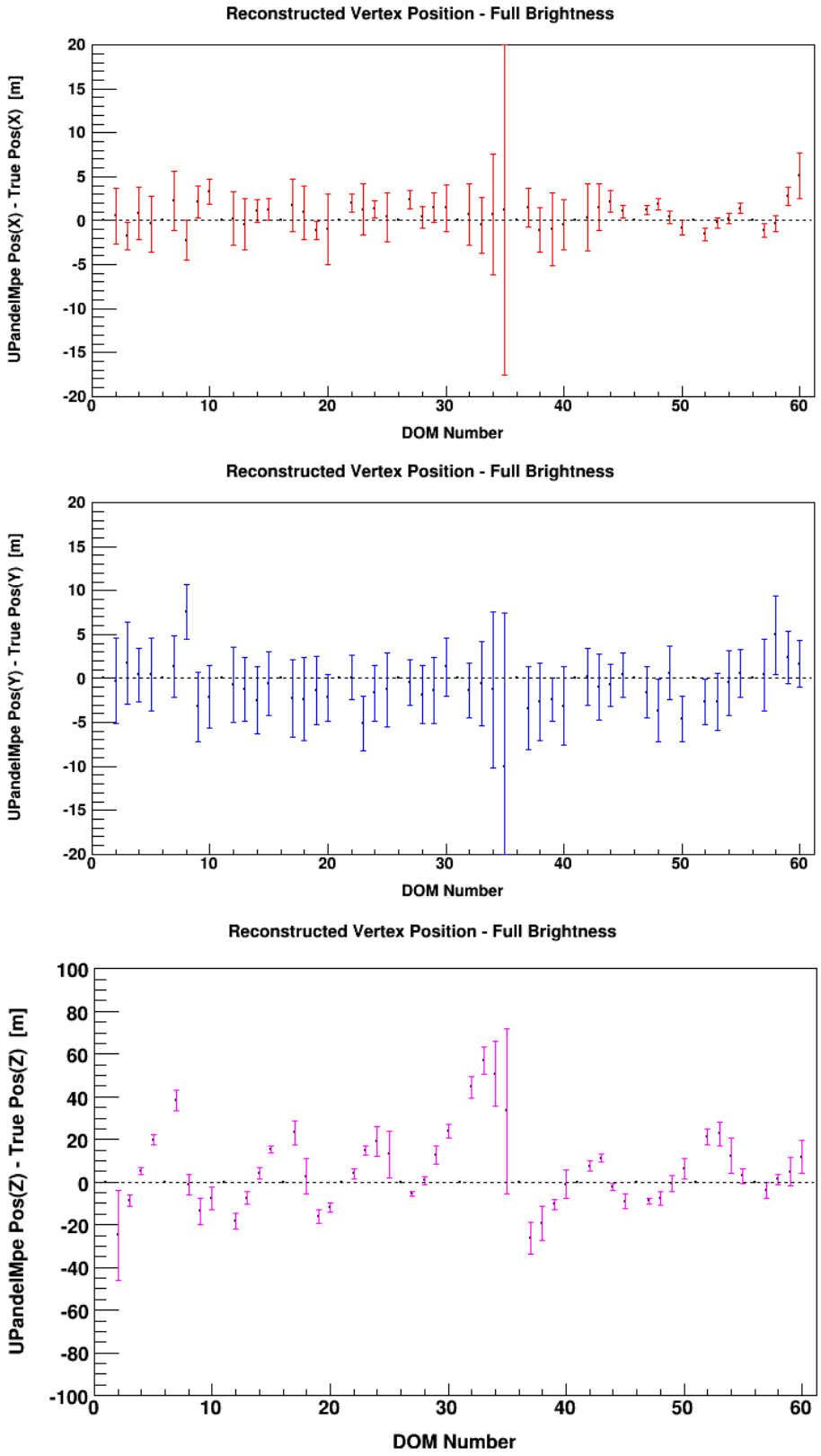

Figure 5.17: (Cscd-llh) UPandelMpe x, y, z vertex position reconstructions for the flasher runs at full LED brightness, with String 63 disabled from readout

5.3.2 Full Brightness Energy/Nph Estimates

The brightness estimates for the analysis with String 63 disabled from readout show wider DOM to DOM distributions on average, due to the wider estimates in the distributions for the reconstruction position of the vertex. Figure 5.7 shows the received total charge difference with the full brightness flasher runs, which displays a wider depth dependent variance with respect to the previous section's analysis.

The wide reconstructed energy distributions are most apparent in the plot showing the estimates for the number of photons produced by the *AtmCscdEnergyReco* module (figure 5.19. As discussed earlier, sensitivity in the utilization of flasher based photorec tables provided to the *AtmCscdEnergyReco* module, causes large sensitivities to the seeded vertex position. Figure 5.19 displays very high (± 1 order of magnitude) RMS values for the estimates of the number of produced photons. The module returns a lower energy estimate for the flasher events near the large dust layer, due to the high offset in reconstructed z position for those events. When seeded with the true vertex position, *AtmCscdEnergyReco* produces a higher than expected estimate for the number of produced photons. This is due to the *AtmCscdEnergyReco* module using a energy conversion which does not take the saturation of DOMs near the light origin into account. With String 63 disabled, the overall saturation effect is of lesser strength. Given the list of disabled DOMs, the *AtmCscdEnergyReco* module uses the expected number of hits on the flashing string, which overcompensates and produces a higher estimate for the number photons than the 'true' value.

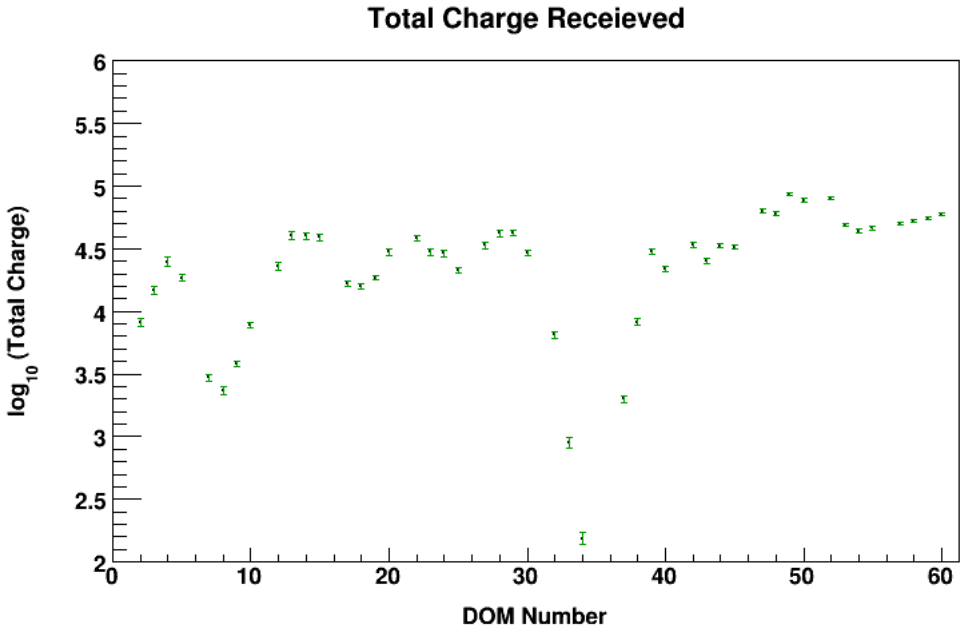

Figure 5.18: Total Charge observed by all of the hit DOMs for the flasher runs at full LED brightness.

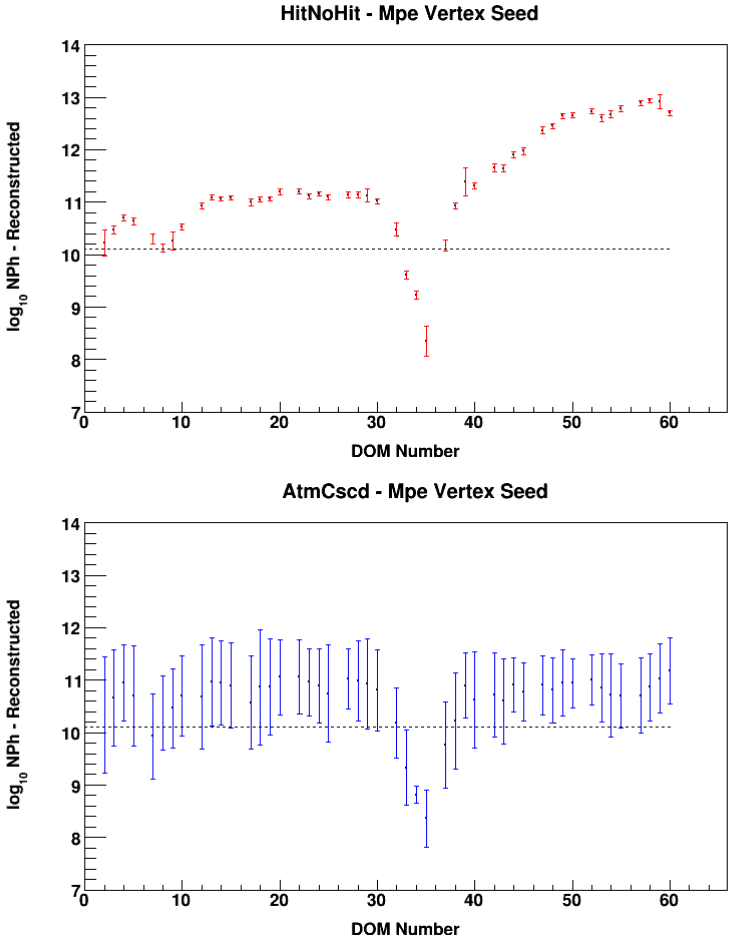

Figure 5.19: NPh (Number of Photons) estimates for each flashing DOM from the UPandelMpe vertex seed for the flasher runs at full LED brightness, with String 63 disabled from readout

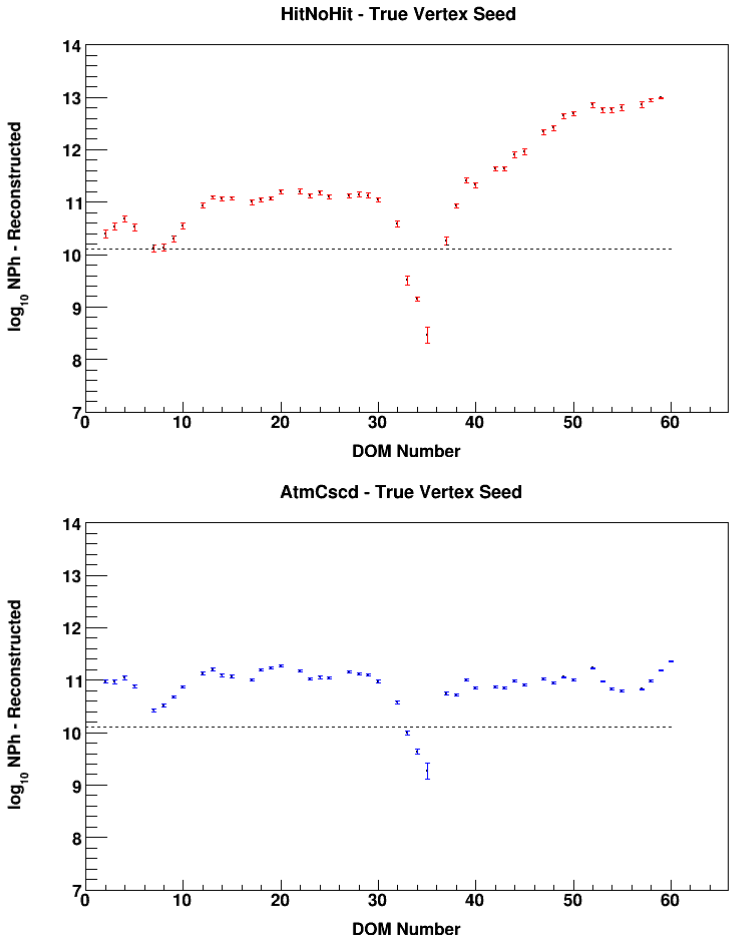

Figure 5.20: NPh (Number of Photons) estimates for each flashing DOM from the True vertex seed for the flasher runs at full LED brightness, with String 63 disabled from readout

5.3.3 Half Brightness Energy/Nph Estimates

Similar to the study done with string 63 enabled in readout, the half brightness runs show very little difference to the full brightness runs in the overall distributions for each of the estimates in figures 5.22 – 5.23. The only notable difference comes from the expected overall shift in the estimates for the produced number of photons, corresponding to the reduced brightness of the LEDs.

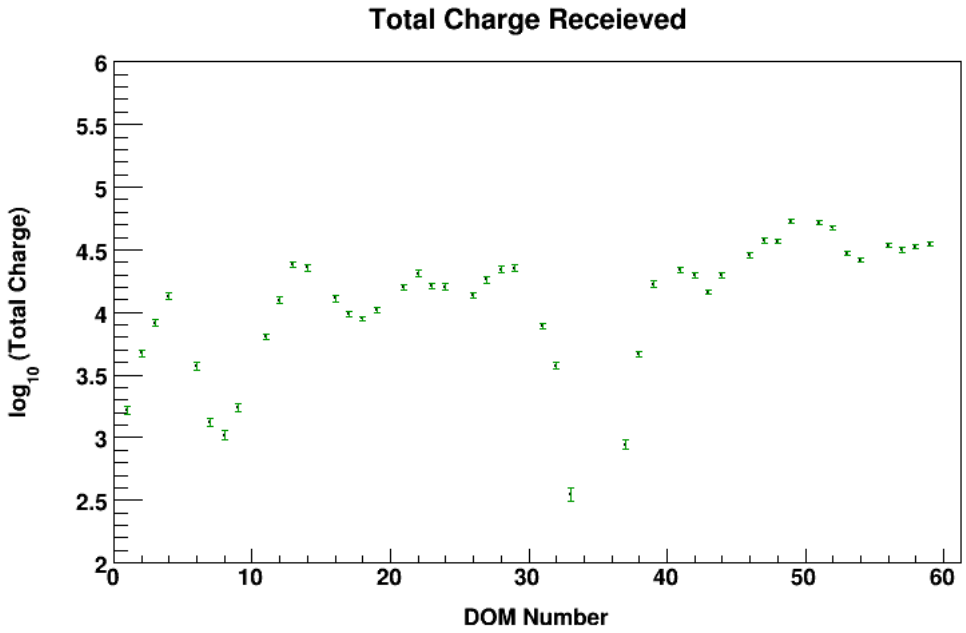

Figure 5.21: Total Charge observed by all of the hit DOMs for the flasher runs at full LED brightness.

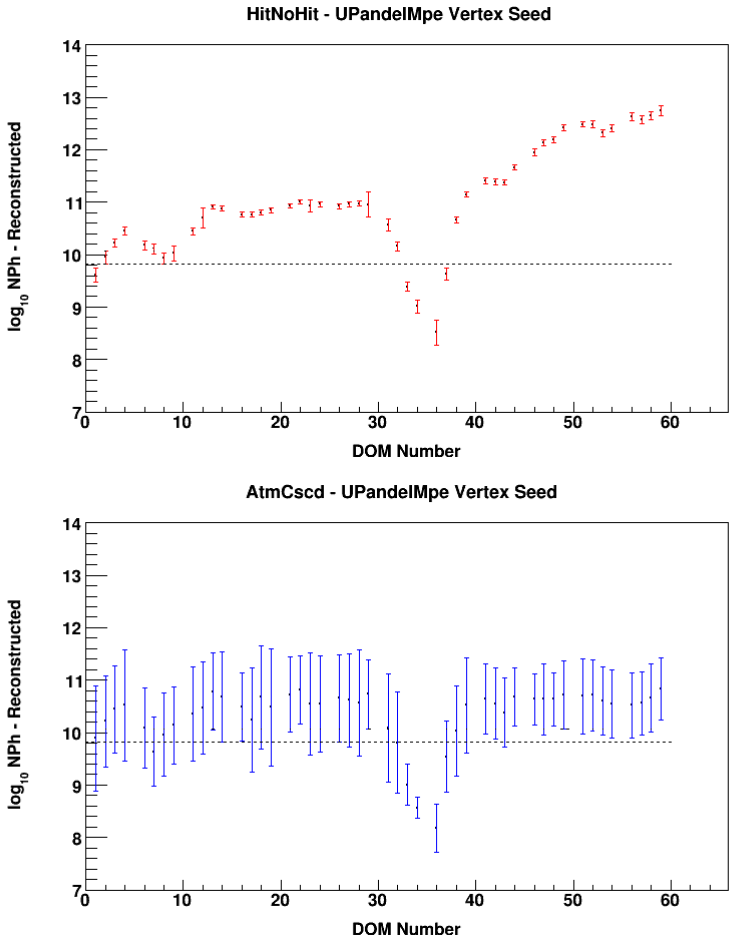

Figure 5.22: NPh (Number of Photons) estimates for each flashing DOM from the UPandelMpe vertex seed for the flasher runs at half LED brightness, with String 63 disabled from readout

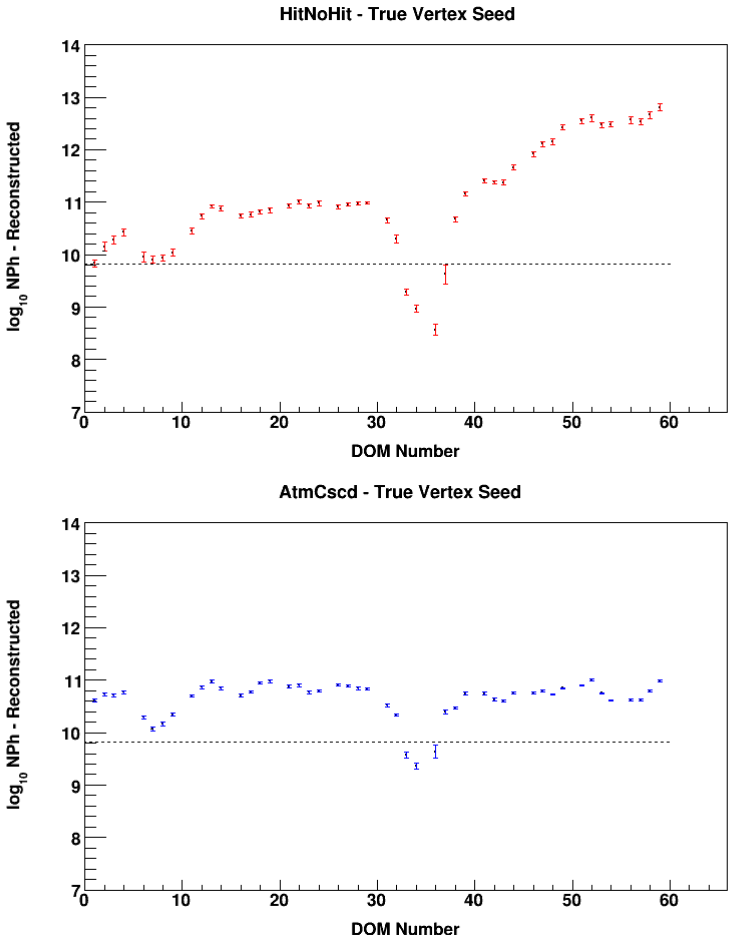

Figure 5.23: NPh (Number of Photons) estimates for each flashing DOM from the True vertex seed for the flasher runs at half LED brightness, with String 63 disabled from readout

6 Conclusion

A study on the reconstruction of cascade like events has been conducted using the IceCube neutrino detector, by using the *in-situ* flasher LEDs on each DOM on String 63. Two different flasher runs were taken in the South Pole each at different brightnesses. Three different reconstruction algorithms were used to estimate the vertex positions, and the number of photons produced by the LEDs located on each DOM for all of these events.

Results for the vertex position estimates reveal problems with the *UPandelMpe* Likelihood Maximizer method, while showing strengths in simple *Center of Gravity* type approaches. This was demonstrated by the events where equal large numbers of hits on DOMs nearest the light origin 'pulled' estimates in the z position reconstruction to their locations. The z estimates often showed a splitting behaviour, due to to the similar hit counts on DOMs nearest the origin. While a having poor z resolution, the *UPandelMpe* routine produced very accurate $x - y$ resolutions. This is also caused from high hit counts on DOMs nearest the origin. A second set of analyses were taken with String 63 disabled from readout, which fixed the splitting behaviour in the z axis, as shown systematically in figure 5.6, at the cost of decreased resolution in the $x-y$ plane. Both brightness settings showed similar overall vertex position estimates.

Results from the number of photon estimates for *Cscdllh* showed a large variation with depth, due to the fact that the module did not take into account scattering and absorption changes at different depths in the detector. The two modules that took these effects into account, *AtmCscdEnergyReco* and *Rime*, displayed invariant averages of the estimates as a function of depth, however with high spread, if seeded with an estimated vertex position. The low average for the reconstructed number of photons produced by the *AtmCscdEnergyReco* module with String63 enabled demonstrated the weaknesses of using flasher based photorec tables to produce accurate estimates when seeded with a reconstructed vertex position. The results for *AtmCscdEnergyReco* improved with String 63 disabled from readout, due to the improved z accuracy for the vertex position estimates provided by the *UPandelMpe* seed.

The study with the two brightness flasher runs can be complimented in the future with data generated by Monte-Carlo simulations. This will enable analysis of the differences between the current ice model used in simulation, and the ice properties surrounding the flashing DOMs. The settings for the reconstruction algorithms could

also be applied to simulated neutrino data, at various positions and energies in the detector. This would show the relative strengths of using specialised module settings for studying flashers, compared with using them on cascade events.

A Appendix

A.1 Code Used for Reconstruction

```
#!/usr/bin/env python

#This script has been built for icerec V02-02-00.

from I3Tray import *

from os.path import expandvars
import os
import sys

load("libdataclasses")
load("libdataio")
load("libphys-services")
load("libDOMcalibrator")
load("libcfirst")
load("libcscd-llh")
load("libFeatureExtractor")
load("liblilliput")
load("libgulliver")
load("libflat-ntuple")
load("libanalysis-tree")
load("libDomTools")
load("librime")
load("libpulse-splitter")
load("libAtmCscdEnergyReco")
load("libsim-services");
load("libsimclasses")
load("libphotonics-service")



true = 1
false = 0

sep_gcd = 0
Reduced_Geo = 0
useBadOMList = 1


DomID = sys.argv[1];


Full_Reconstruction = 1

filename = "Run001117xx_bright127_DOM"+str(DomID)+"-flashersonly"

workspace = expandvars("$I3_SRC")
```

```
48 tools = expandvars("$I3_TOOLS")
49 outdir = os.path.join("/net/user/jmccartin/exp-data/Run1117xx/reco/")
50 datadir = os.path.join("/net/user/jmccartin/exp-data/Run1117xx/sorted_byDOM/
       bright127/")
51 datafile = datadir+filename+".i3"
52
53
54 runTag = datafile
55 rootfilename = outdir+"2RECO-AC-FLASHSTRING_OFF-"+filename+".root"
56 ntuplerootfilename = outdir+"2RECO-AC-FLASHSTRING_OFF-"+filename+".ntuple.root
       "
57 i3outfilename = outdir+"2RECO-AC-FLASHSTRING_OFF-"+filename+".i3"
58
59 mbids = workspace + "/phys-services/resources/doms.txt"
60 amageofile = workspace + "/phys-services/resources/amanda.geo"
61 icegeofile = workspace + "/phys-services/resources/icecube.geo"
62 pulseClean1 =      "pulseClean1"
63 pulseClean2 =      "pulseClean2"
64 pulseClean3 =      "pulseClean3"
65 hit_map =         "hit_map"
66
67 AnalysisSeries = "FEPulses"
68
69
70
71 pulse_map = pulseClean2
72
73 ExStrings = [21, 29, 30, 38, 39, 40, 47, 48, 49, 50, 52, 57, 58, 59, 60, 66,
       67, 68, 73, 74, 75]
74
75 ZSeeds = [0, 496.72, 479.7, 462.68, 445.66, 428.64, 411.62, 394.6, 377.58,
       360.56, 343.54, 326.52, 309.5, 292.48, 275.45, 258.43, 241.41, 224.39,
       207.37, 190.35, 173.33, 156.31, 139.29, 122.27, 105.25, 88.23, 71.21,
       54.19, 37.16, 20.14, 3.12, -13.9, -30.92, -47.94, -64.96, -81.98, -99,
       -116.02, -133.04, -150.06, -167.08, -184.11, -201.13, -218.15, -235.17,
       -252.19, -269.21, -286.23, -303.25, -320.27, -337.29, -354.31, -371.33,
       -388.35, -405.38, -422.4, -439.42, -456.44, -473.46, -490.48, -507.5]
76
77 XSeed = -66.700
78 YSeed = 276.920
79 ZSeed = ZSeeds[int(DomID)]
80
81 print "Reconstructing flasher events for DOMID = "+str(DomID)+"( Z = "+str(
       ZSeed)+" )"
82
83 if useBadOMList:
84   if int(DomID)<06:
85     i3_BadOMList = [OMKey(21, 30),
86             OMKey(63, int(DomID)) ,   #Excludes the flashing DOMs in the run
87             OMKey(63, (int(DomID)+5)) ,   #Excludes the flashing DOMs in the
                    run
88             OMKey(29, 59) ,
89             OMKey(29, 60) ,
90             OMKey(30, 23) ,
91             OMKey(30, 60) ,
92             OMKey(38, 59) ,
93             OMKey(39,  8) ,
94             OMKey(39, 22) ,
95             OMKey(40, 51) ,
96             OMKey(40, 52) ,
97             OMKey(46, 49) ,
```

```
98              OMKey(46, 50) ,
99              OMKey(46, 53) ,
100             OMKey(46, 54) ,
101             OMKey(46, 55) ,
102             OMKey(46, 56) ,
103             OMKey(47, 55) ,
104             OMKey(47, 56) ,
105             OMKey(50, 36) ,
106             OMKey(50, 58) ,
107             OMKey(56, 58) ,
108             OMKey(58, 45) ,
109             OMKey(72, 33) ,
110             OMKey(72, 34) ,
111             OMKey(72, 35) ,
112             OMKey(72, 36) ,
113             OMKey(72, 37) ,
114             OMKey(72, 38) ,
115             OMKey(72, 43) ,
116             OMKey(74,  9) ]
117   if 05<int(DomID)<55:
118     i3_BadOMList = [OMKey(21, 30),
119                             OMKey(63, (int(DomID)+5)) ,
120             OMKey(63, (int(DomID)-5)) ,   #Excludes the flashing DOMs in the
                    run
121             OMKey(63, int(DomID)) ,   #Excludes the flashing DOMs in the run
122             OMKey(29, 59) ,
123             OMKey(29, 60) ,
124             OMKey(30, 23) ,
125             OMKey(30, 60) ,
126             OMKey(38, 59) ,
127             OMKey(39,  8) ,
128             OMKey(39, 22) ,
129             OMKey(40, 51) ,
130             OMKey(40, 52) ,
131             OMKey(46, 49) ,
132             OMKey(46, 50) ,
133             OMKey(46, 53) ,
134             OMKey(46, 54) ,
135             OMKey(46, 55) ,
136             OMKey(46, 56) ,
137             OMKey(47, 55) ,
138             OMKey(47, 56) ,
139             OMKey(50, 36) ,
140             OMKey(50, 58) ,
141             OMKey(56, 58) ,
142             OMKey(58, 45) ,
143             OMKey(72, 33) ,
144             OMKey(72, 34) ,
145             OMKey(72, 35) ,
146             OMKey(72, 36) ,
147             OMKey(72, 37) ,
148             OMKey(72, 38) ,
149             OMKey(72, 43) ,
150             OMKey(74,  9) ]
151   if int(DomID)>54:
152     i3_BadOMList = [OMKey(21, 30),
153             OMKey(63, (int(DomID)-5)) ,   #Excludes the flashing DOMs in the
                    run
154             OMKey(63, int(DomID)) ,   #Excludes the flashing DOMs in the run
155             OMKey(29, 59) ,
156             OMKey(29, 60) ,
```

```
157             OMKey(30, 23) ,
158             OMKey(30, 60) ,
159             OMKey(38, 59) ,
160             OMKey(39,  8) ,
161             OMKey(39, 22) ,
162             OMKey(40, 51) ,
163             OMKey(40, 52) ,
164             OMKey(46, 49) ,
165             OMKey(46, 50) ,
166             OMKey(46, 53) ,
167             OMKey(46, 54) ,
168             OMKey(46, 55) ,
169             OMKey(46, 56) ,
170             OMKey(47, 55) ,
171             OMKey(47, 56) ,
172             OMKey(50, 36) ,
173             OMKey(50, 58) ,
174             OMKey(56, 58) ,
175             OMKey(58, 45) ,
176             OMKey(72, 33) ,
177             OMKey(72, 34) ,
178             OMKey(72, 35) ,
179             OMKey(72, 36) ,
180             OMKey(72, 37) ,
181             OMKey(72, 38) ,
182             OMKey(72, 43) ,
183             OMKey(74,  9) ]
184   i3_bad_count = len( i3_BadOMList )
185
186   #
187   #  ..................... 2nd: List of IceCube strings and then DOMs that do
          not exist yet
188   #
189
190   list_of_inactive_strings_ic40 = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12,
          13, 14, 15, 16, 17, 18, 19, 20, 22, 23, 24, 25, 26, 27, 28, 31, 32, 33,
          34, 35, 36, 37, 41, 42, 43, 51, 79, 80]
191   i3_Nonexistent_DOMList  = [ ]
192   for string_no in list_of_inactive_strings_ic40 :
193     for om_no in range(60) :                          # 0 ... 59,  so do +1
            below
194       this_omkey = OMKey( string_no ,  om_no+1 )
195       i3_Nonexistent_DOMList.append( this_omkey )
196
197   #
198   #  ..................... 3rd: set up the IceTop excluded OM list
199   #
200
201   #
202   #  ..................... 4th: set up the I3 excluded OM list (for use with I3
          -only fits)
203   #
204
205   i3_ExcludedOMList = i3_BadOMList                    # Erik's I3 bad DOM list
206   i3_ExcludedOMList.extend( i3_Nonexistent_DOMList ) # + remove nonexistent I3
          DOM list for IC40
207   for string_no in [ -20 ,  -19 ,  -18 ,  -17 ,  -16 ,  -15 ,  -14 ,  -13 ,
          -12 ,  -11 ,
208               -10 ,  -9 ,  -8 ,  -7 ,  -6 ,  -5 ,  -4 ,  -3 ,  -2 ,  -1
                    ] :  #Add '63' to disable String 63 from readout
209     for om_no in range(60) :                        # 0 ... 59,  so do +1 below
```

```
        this_omkey = OMKey( string_no , om_no+1 )
        i3_ExcludedOMList.append( this_omkey )


tray = I3Tray()

##########################################################################
### SERVICES
##########################################################################

tray.AddService("I3ReaderServiceFactory", "reader")(
  ("filename", datafile))


tray.AddService("I3RootTreeServiceFactory", "service")(
  ("TreeFileName", rootfilename)
  )

if Reduced_Geo:
  StringsToUse = "44:46, 53:56, 61:65, 69:72, 76:78"
  tray.AddService("I3GeometrySelectorServiceFactory", "geo-selector")(
    ("StringsToUse", StringsToUse),
    ("StationsToExclude", "-19:80"),
    #("ShiftToCenter", True),
    ("GeoSelectorName", "IC40-NoTop-Geo")
    )

driverdir  = expandvars("$I3_SRC")
driverfile = "level1_flasher_simulation.list"
driverfile_l2 = "level2_flasher_photorec.list"
tabledir   = expandvars("$I3_SRC")+"/tables/"


tray.AddService("I3PhotonicsServiceFactory", "photonics-service")(
    ("TablesDirectory", driverdir),
    ("DriverFileDirectory", driverdir),
    ("PhotonicsTableSelection", 2),
    #("PhotonicsLevel1DriverFile", driverfile),
    ("PhotonicsLevel2DriverFile", driverfile_l2),
    ("UseDummyService", False)
    )



##########################################################################
### MODULES
##########################################################################
if Reduced_Geo:
  tray.AddModule("I3Muxer",  "muxer")(
    ("GeometryService", "IC40-NoTop-Geo")
    )

else:
  tray.AddModule("I3Muxer",  "muxer")(
    )


tray.AddModule("I3DOMLaunchCleaning", "launchcleaning")(
  ("InIceInput", "InIceRawData"),  #DOMLaunchSeriesMap
  ("InIceOutput", "CleanInIceRawData"),
  ("FirstLaunchCleaning", True),
```

```
271     ("CleanedKeys", i3_ExcludedOMList)
272     )
273
274
275 tray.AddModule("I3DOMcalibrator", "domcalibrator")(
276     ("InputRawDataName", "CleanInIceRawData"),
277     ("OutputATWDDataName", "CalibratedATWD"),
278     ("OutputFADCDataName", "CalibratedFADC")
279     )
280
281 tray.AddModule("I3FeatureExtractor", "features")(
282     ("InitialHitSeriesReco", "FEHits"),
283     ("InitialPulseSeriesReco", "FEPulses"),
284     ("CalibratedFADCWaveforms", "CalibratedFADC"),
285     ("CalibratedATWDWaveforms", "CalibratedATWD"),
286     ("RawReadoutName", "CleanInIceRawData"),
287     ("MaxNumHits", 0),
288     ("FastFirstPeak", 11),
289     ("FastPeakUnfolding", 0 ),
290     )
291
292 ####################################
293 ##          Vertex Seeder
294 ####################################
295
296 tray.AddModule("PutVertexModule", "VertexSeeder")(
297       ("VertexName", "FlasherLocation"),
298       ("XPos", XSeed),
299       ("YPos", YSeed),
300       ("ZPos", ZSeed)
301       )
302
303 ####################################
304 ##        First Guess Method
305 ####################################
306
307
308 tray.AddModule("I3CFirstModule", "CFirst")(
309     ("InputType", "RecoPulse"),
310     ("RecoSeries", "FEPulses"),
311     ("MinHits", 10),
312     ("TriggerHits", 3),
313     ("TriggerWindow", 100),
314     ("EarlyHitStart", -3300),
315     ("EarlyHitStop", -200),
316     ("DirectHitRadius", 1000),
317     ("AmpWeightPower", 0.0),
318     ("SmallClusterRadius", 1000),
319     ("LargeClusterRadius", 1000),
320     ("FirstLE", False),
321     ("ResultName", "CFirst")
322     )
323
324
325 ####################################
326 ## Cascade Reconstruction Modules #
327 ####################################
328
329 if Full_Reconstruction:
330     tray.AddModule("I3CscdLlhModule", "vertex-cscdllh")(
331       ("InputType", "RecoPulse"),
```

```
332     ("RecoSeries", "FEPulses"),
333     ("SeedKey", "CFirst"),
334     ("ResultName", "VertexReco_UPandel"),
335     ("MinHits", 8),
336     ("Minimizer", "Powell"),
337     ("PDF", "UPandel"),
338     ("ParamT", "1.0, 0.0, 0.0, false"),
339     ("ParamX", "1.0, 0.0, 0.0, false"),
340     ("ParamY", "1.0, 0.0, 0.0, false"),
341     ("ParamZ", "1.0, 0.0, 0.0, false"),
342     ("FirstLE", True),
343     ("AmpWeightPower", 1.0),
344     ("PandelSmallProb", 1.0e-6),
345     ("PandelTau", 450.0),
346     ("PandelLambda", 47.0),
347     ("PandelLambdaA", 98.0),
348     ("PandelSigma", 15.0),
349     ("PandelMaxDist", 500.0),
350     )
351
352   # SPE fit from the cascade fitter vertex-cscdllh (previous module)
353   tray.AddModule("I3CscdLlhModule", "mpe-vertex-cscdllh")(
354     ("InputType", "RecoPulse"),
355     ("RecoSeries", "FEPulses"),
356     ("SeedKey", "VertexReco_UPandel"),
357     ("ResultName", "VertexReco_UPandelMpe"),
358     ("MinHits", 8),
359     ("Minimizer", "Powell"),
360     ("PDF", "UPandelMpe"),
361     ("ParamT", "1.0, 0.0, 0.0, false"),
362     ("ParamX", "1.0, 0.0, 0.0, false"),
363     ("ParamY", "1.0, 0.0, 0.0, false"),
364     ("ParamZ", "1.0, 0.0, 0.0, false"),
365     ("FirstLE", True),
366     ("AmpWeightPower", 1.0),
367     ("PandelSmallProb", 1.0e-6),
368     ("PandelTau", 450.0),
369     ("PandelLambda", 47.0),
370     ("PandelLambdaA", 98.0),
371     ("PandelSigma", 15.0),
372     ("PandelMaxDist", 500.0),
373     )
374
375
376   ################
377   ##  HitNoHit
378   ################
379
380
381   ## Energy reco using mpe-vertex-cscdllh as its seed
382   tray.AddModule("I3CscdLlhModule", "standardenergyreco")(
383     ("InputType", "RecoPulse"),
384     ("RecoSeries", "FEPulses"),
385     ("SeedKey", "VertexReco_UPandelMpe"),
386     ("ResultName", "EnergyReco_HitNoHit"),
387     ( "MinHits", 8 ),
388     ( "FirstLE", True ),
389     ( "Minimizer", "Brent" ),
390     ( "EnergySeed", 5.4 ),
391     ( "AmpWeightPower", 0.0 ),
392     ( "PDF", "HitNoHit"),
```

```
393       ("ParamX", "0.0, 0.0, 0.0, true"),
394       ("ParamY", "0.0, 0.0, 0.0, true"),
395       ("ParamZ", "0.0, 0.0, 0.0, true"),
396       ("ParamEnergy", "22., 2.3, 8.5, false"),
397       ("ExcludedOMs", i3_ExcludedOMList),
398       ("MinimizeInLog(E)", True)
399       )
400
401
402   ###############
403   ##  HitNoHit Seeded
404   ###############
405
406
407   # Energy reco using mpe-vertex-cscdllh as its seed
408   tray.AddModule("I3CscdLlhModule", "standardenergyreco2")(
409     ("InputType", "RecoPulse"),
410     ("RecoSeries", "FEPulses"),
411     ("SeedKey", "FlasherLocation"),
412     ("ResultName", "EnergyReco_HitNoHit_TrueSeed"),
413     ("MinHits", 8),
414     ("FirstLE", True),
415     ("Minimizer", "Brent"),
416     ("EnergySeed", 5.4),
417     ("AmpWeightPower", 0.0),
418     ("PDF", "HitNoHit"),
419     ("ParamX", "0.0, 0.0, 0.0, true"),
420     ("ParamY", "0.0, 0.0, 0.0, true"),
421     ("ParamZ", "0.0, 0.0, 0.0, true"),
422     ("ParamEnergy", "22., 2.3, 8.5, false"),
423     ("ExcludedOMs", i3_ExcludedOMList),
424     ("MinimizeInLog(E)", True)
425     )
426
427
428   ###############
429   ##  AtmCscd
430   ###############
431
432   tray.AddModule("AtmCscdEnergyReco", "photorec-energy")(
433     ("Name", "EnergyReco_AtmCscdMpe"),
434     ("InputRecoPulses", "FEPulses"),
435     ("CascadeVertex", "VertexReco_UPandelMpe"),
436     ("PhotonicsServiceName", "PhotonicsService"),
437     ("CascadeVertexZenithAngle", 0.0),
438     ("IgnoreAMANDA", True),
439     ("SaveAllLlhTerms", False),
440     ("WriteDiagnosticRootFile", False),
441     ("KeysToClean", i3_ExcludedOMList)
442     )
443
444   tray.AddModule("AtmCscdEnergyReco", "photorec-energy-seeded")(
445     ("Name", "EnergyReco_AtmCscdMpe_TrueSeed"),
446     ("InputRecoPulses", "FEPulses"),
447     ("CascadeVertex", "FlasherLocation"),
448     ("PhotonicsServiceName", "PhotonicsService"),
449     ("CascadeVertexZenithAngle", 0.0),
450     ("IgnoreAMANDA", True),
451     ("SaveAllLlhTerms", False),
452     ("WriteDiagnosticRootFile", False),
453     ("KeysToClean", i3_ExcludedOMList)
```

```
454       )
455
456
457
458     ################
459     ##   Rime
460     ################
461
462     tray.AddModule("I3Rime",   "rime")(
463       ("RecoTrack",   1),
464       ("RecoInIceWeightDOMs", 1),
465       ("RecoKeepAll", 0),
466       ("RecoJoint",   1),
467       ("RecoCascade", 1),
468       ("Verbose",   0),
469       ("RecoString",   0),
470       ("ParamsSuffix",   "Params"),
471       ("RecoPulseSeriesNames",   ["FEPulses"]),
472       ("RecoName", "Rime")
473       )
474
475
476
477
478 ##########################################
479 ## End of Cascade Reconstruction Modules #
480 ##########################################
481
482 ###################################
483 ##    FLAT-NTUPLE    ROOT CREATOR ##
484 ###################################
485
486 tray.AddModule("I3FlatNtupleModule", "pawRoolz")(
487     ("BookNDirect", True),
488     ("BookOMs", True),
489     ("BookSkyCoords", True),
490     ("BookTracks", True),
491     ("BookUserLines", True),
492     ("DOMLaunch", "CleanInIceRawData"),
493     ("EventHeader", "I3EventHeader"),
494     ("FavoriteFit",   "VertexReco_UPandelMpe"),
495     ("FavoritePulses",   "FEPulses"),
496     ("FavoriteHits", "FEHits"),
497     ("FavoriteReadOut", ""),
498     ("HitSelection", ""),
499     ("MCTruthName", "I3MCTree"),
500     ("TriggerLabel", "I3TriggerHierarchy"),
501     ("MCWeightsName", "I3MCWeightDict"),
502     ("OutFile", ntuplerootfilename),
503     ("TreeName", "Reconstruction"),
504     )
505
506 ###################################
507 ##    ANALYSIS-TREE ROOT CREATOR ##
508 ###################################
509
510 tray.AddModule("I3RootTreeModule", "recotree")(
511     )
512 tray.AddModule("I3HitsTreeModule<I3RecoHit>", "hitstree")(
513     ("KeyNames",   AnalysisSeries)
514     )
```

```
515 tray.AddModule("I3DOMTreeModule<I3RecoHit>", "domtree")(
516     ("KeyNames",    AnalysisSeries)
517     )
518 tray.AddModule("I3DOMTreeModule<I3DOMLaunch>", "DOMLaunchTree")(
519     ("KeyNames", "InIceFlasher")
520     )
521 tray.AddModule("I3EventInfoTreeModule<I3RecoHit>", "eventinfo")(
522     ("KeyNames",    AnalysisSeries)
523     )
524 tray.AddModule("I3ResidualsTreeModule", "res")(
525     ("HitsName",    AnalysisSeries),
526     ("ParticleNames", "SimpleReco")
527     )
528
529 ####################################
530 ##          I3 FILE WRITER        ##
531 ####################################
532
533 tray.AddModule("I3Writer", "writer")(
534         ("filename", i3outfilename),
535         ("SkipKeys", ["InIceRawData", "CleanIceTopRawData", "CleanInIceRawData",
                "allHitsDirty", "allHits"])
536         )
537
538 tray.AddModule("TrashCan",    "the can")
539 tray.Execute()
540 #tray.Execute(int(nevents)+4)
541 tray.Finish()
```

Listing A.1: Python steering file used to run the reconstruction methods.

Bibliography

[1] Kan Chang Wang. A suggestion on the detection of the neutrino. *Phys. Rev.*, 61(1-2):97, Jan 1942.

[2] C. L. Cowan, Jr., F. Reines, F. B. Harrison, H. W. Kruse, and A. D. McGuire. Detection of the Free Neutrino: A Confirmation. *Science*, 124:103–104, July 1956.

[3] The Royal Swedish Academy of Sciences. Nobel prize in physics. *nobelprize.org:/physics/1995*, 1995.

[4] G. Danby, J-M. Gaillard, K. Goulianos, L. M. Lederman, N. Mistry, M. Schwartz, and J. Steinberger. Observation of high-energy neutrino reactions and the existence of two kinds of neutrinos. *Phys. Rev. Lett.*, 9(1):36–44, Jul 1962.

[5] ML Perl, GS Abrams, AM Boyarski, M. Breidenbach, DD Briggs, F. Bulos, W. Chinowsky, JT Dakin, GJ Feldman, CE Friedberg, et al. Evidence for Anomalous Lepton Production in e^{+}-e^{-} Annihilation. *Physical Review Letters*, 35(22):1489–1492, 1975.

[6] K. Kodama, N. Ushida, C. Andreopoulos, N. Saoulidou, G. Tzanakos, P. Yager, B. Baller, D.J. Boehnlein, W. Freeman, B. Lundberg, et al. Observation of tau neutrino interactions. *Physics Letters B*, 504(3), 2001.

[7] Y. Fukuda, T. Hayakawa, E. Ichihara, K. Inoue, K. Ishihara, H. Ishino, Y. Itow, T. Kajita, J. Kameda, S. Kasuga, K. Kobayashi, Y. Kobayashi, Y. Koshio, M. Miura, M. Nakahata, S. Nakayama, A. Okada, K. Okumura, N. Sakurai, M. Shiozawa, Y. Suzuki, Y. Takeuchi, Y. Totsuka, S. Yamada, M. Earl, A. Habig, and E. Kearns. Evidence for oscillation of atmospheric neutrinos. *Phys. Rev. Lett.*, 81(8):1562–1567, Aug 1998.

[8] S.P. Mikheev and A. Yu. Smirnov. *Sov. J. Nucl. Phys.*, 42:913, 1985.

[9] Q. R. Ahmad, R. C. Allen, T. C. Andersen, J. D. Anglin, G. Bühler, J. C. Barton, E. W. Beier, M. Bercovitch, J. Bigu, S. Biller, R. A. Black, I. Blevis, R. J. Boardman, J. Boger, E. Bonvin, M. G. Boulay, M. G. Bowler, T. J. Bowles, S. J. Brice, M. C. Browne, T. V. Bullard, T. H. Burritt, K. Cameron, J. Cameron, Y. D. Chan, M. Chen, and H. H. Chen. Measurement of the rate of $\nu e + d \rightarrow$

$p + p + e-$ interactions produced by $b8$ solar neutrinos at the sudbury neutrino observatory. *Phys. Rev. Lett.*, 87(7):071301, Jul 2001.

[10] The Royal Swedish Academy of Sciences. Nobel prize in physics. *nobelprize.org:/physics/2002*, 2002.

[11] S. Klein et al. IceCube: A Cubic Kilometer Radiation Detector. *Lawrence Berkeley National Laboratory*, 2008.

[12] M. Ackermann, J. Ahrens, X. Bai, M. Bartelt, SW Barwick, RC Bay, T. Becka, JK Becker, K. Becker, and P. Berghaus. Optical properties of deep glacial ice at the South Pole (DOI 10.1029/2005JD006687). *JOURNAL OF GEOPHYSICAL RESEARCH-ALL SERIES-*, 111(D13):13203, 2006.

[13] K. Woschnagg et al. Optical Properties of South Pole Ice at Depths from 140 to 2300 Meters. *Proceedings of the 26th International Cosmic Ray Conference*, 2:200–203.

[14] P. Askebjer, SW Barwick, L. Bergstrom, A. Bouchta, S. Carius, A. Coulthard, K. Engel, B. Erlandsson, A. Goobar, L. Gray, et al. Optical Properties of the South Pole Ice at Depths Between 0.8 and 1 km. *Arxiv preprint astro-ph/9412028*, 1994.

[15] S. Hickford. Simulation of cascades for the icecube neutrino telescope. *MSc Thesis, 2007*.

[16] R. Gandhi, C. Quigg, M. Hall Reno, and I. Sarcevic. Ultrahigh-energy neutrino interactions. *Astroparticle Physics*, 5(2):81–110, 1996.

[17] I. Fermin. Search for high energy neutrino induced cascades with the *AMANDA B10* detector. *PhD Thesis, 2002*.

[18] M.L. Perl. Tau missing-decay-mode problem and limits on a second tau neutrino. *Physical Review D*, 38(3):845–849, 1988.

[19] T. DeYoung, S. Razzaque, and DF Cowen. A novel tau signature in neutrino telescopes. In *Journal of Physics: Conference Series*, volume 60, pages 231–234. Institute of Physics Publishing, 2007.

[20] Shigeru Yoshida, Rie Ishibashi, and Hiroko Miyamoto. Propagation of extremely high energy leptons in earth: Implications for their detection by the icecube neutrino telescope. *Phys. Rev. D*, 69(10):103004, May 2004.

[21] LD Landau and IJ Pomeranchuk. â œElectron-cascade processes at ultra-high energies, â Dokl. *Akad. Nauk SSSR*, 92:735, 1953.

[22] J. Ahrens, X. Bai, R. Bay, SW Barwick, T. Becka, JK Becker, K.H. Becker, E. Bernardini, D. Bertrand, A. Biron, et al. Muon track reconstruction and

data selection techniques in AMANDA. *Nuclear Inst. and Methods in Physics Research, A*, 524(1-3):169–194, 2004.

[23] Sean Grullon. Tensor of inertia module documentation. *http://www.ifh.de/ kislat/www/icerec/docs/trunk/tensor-of-inertia/index.html.*

[24] Michael Greene. CFirst module documentation. *http://www.ifh.de/ kislat/www/icerec/docs/trunk/cfirst/index.html.*

[25] D. Pandel. Bestimmung von Wasser-und Detektorparametern und Rekonstruktion von Myonen bis 100 TeV mit dem Baikal-Neutrinoteleskop NT-72.

[26] Michael Greene. Cscd-llh module documentation. *http://www.ifh.de/ kislat/www/icerec/docs/trunk/cscd-llh/index.html.*

[27] D. Chirkin. Rime module documentation. *http://www.ifh.de/ kislat/www/icerec/docs/trunk/rime/index.html.*

[28] D. Chirkin. Rime Module Presentation. *Baton-Rouge Collaboration Meeting, 2006.*

[29] D. Chirkin. Flasher Reconstruction Presentation (slide 15). *London Collaboration Meeting, 2006.*

Acknowledgements

I would like to thank my supervisor, Dr. Jenni Adams for her help and guidance throughout my thesis, and providing me with the chance to go and visit fellow IceCube collaborators in Berkeley, California. Thanks goes to Dr. Spencer Klein for offering the invitation for this unique opportunity. I would also like to acknowledge Dr. Surujhdeo Seunarine and Dr. Andreas Gross for their help with the explaining of certain technical aspects behind the reconstruction methods, and their assistance with the write-up process.

A very special thanks go to my fellow IceCube cascade collaborators who helped me with the execution, configuration and analysis of the flasher runs used in this study. Many thanks especially goes to Matthias Danniger and Michelangelo D'Agostino for their assistance and paitience while helping me understand the details of the Python and ROOT languages used in the software analysis, and the configuration of the *IceTray* reconstruction modules.

Final thanks goes to Giles Reid and Richard Graham for proof reading, and assistance with the Linux operating system, of which I could have not produced this thesis without.